Lecture Notes in Computer Science 2856

Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Michael Smirnov Ernst Biersack Chris Blondia
Olivier Bonaventure Olga Casals
Gunnar Karlsson George Pavlou Bruno Quoitin
James Roberts Ioannis Stavrakakis
Burkhard Stiller Panos Trimintzios
Piet Van Mieghem (Eds.)

Quality of Future Internet Services

COST Action 263 Final Report

Springer

Volume Editor

Michael Smirnov
Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31, 10589 Berlin
Germany
E-mail: smirnov@fokus.fraunhofer.de

Cataloging-in-Publication Data applied for

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie;

detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

CR Subject Classification (1998): C.2, H.4, H.3, H.5, D.2, K.6

ISSN 0302-9743
ISBN 3-540-20193-9 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag Berlin Heidelberg New York
a member of BertelsmannSpringer Science+Business Media GmbH

http://www.springer.de

Typesetting: Camera-ready by author, data conversion by Boller Mediendesign
Printed on acid-free paper SPIN 10958421 06/3142 5 4 3 2 1 0

Preface

This state-of-the-art survey of technologies, algorithms, models, and experiments in the area of Internet Quality of Service is the final report of COST (European Cooperation in the field of Scientific and Technical Research) Action 263, *Quality of future Internet Services, http://www.fokus.fraunhofer.de/cost263* (QofIS). COST 263 ran from January 1999 until October 2003 with the participation of some 70 researchers from 38 organizations from 14 European countries (France, Hungary, Italy, Sweden, UK, Belgium, Germany, the Netherlands, Portugal, Slovenia, Switzerland, Finland, Greece, Romania, and Spain). The Action belonged to the COST area "Multimedia and Internet Communication"; together with COST 264, *Networked Group Communication*, this Action continued the effort started in 1992 by COST 237, *Multimedia Telecommunications Services*. Both groups have combined their efforts now in the Network of Excellence in *Emerging Networking Experiments and Technologies, http://www.ist-e-next.net* (E-NEXT) of the 6th European Framework program.

The book consists of seven chapters that were written in 18 months by 67 individual authors. The main objective of this book is to report the state of the art in the area of QofIS, as percieved by the authors, including achievements that were originated by the authors. The book was designed in a top-down manner: after three years of running the Action with close co-ordination of research efforts, it was easy to achieve a consensus on the table of contents. To ensure the content quality the following roles were defined and assigned to COST 263 members: chapter editor, chapter author, chapter reader. In all cases the chapter editor was also a chapter author, while the chapter reader was selected from among the non-authors for that chapter. Our main method of working was electronic mail and shared workspace powered by the *Basic Support for Cooperative Work http://bscw.fokus.fraunhofer.de* (BSCW) server.

After that, groups of authors contributed individual papers to particular chapters. Chapter editors did the initial integration of the papers, providing comments to the authors, and enforcing improvements that often came from the chapter readers. Actually, it took longer than expected; however, the COST 263 community was committed to continuing this effort, and in June 2002 the Action applied for a one-year extension of COST 263 with the particular objective of finishing this book. We thankfully acknowledge COST TC-TIST and COST CSO for the approval of this extension. The final integration step was done by the chapter editors – all individual contributions are now presented within a single flow of text with a common introduction, conclusion and list of references per chapter. The first chapter aims at providing a roadmap for the whole book and serves as a summary of the book's content.

We wish to record our appreciation of the efforts of many people who helped to make this book happen: to the authors (we regret that it was not possible to integrate all the individual contributions); to the COST TC-TIST members and

the Secretariat, originally part of the European Commission and lately part of the European Science Foundation; and to Springer-Verlag for continuous support of the QofIS workshop *http://www.qofis.org* and this final report.

We learned about the untimely death of our dear colleague Prof. Olga Casals on June 11 and we would like to dedicate this volume to her memory.

July 2003 Michael Smirnov
Berlin (on behalf of all the editors)

About This Book

This book is the final report of COST – *Cooperation europeenne dans le domaine de la recherche scientifique et technique*, Action 263 – *Quality of future Internet Services.*

Editors

Book Editor	Smirnov, M. (Germany)
Chapter Editors	Biersack, E. (France)
	Blondia, C. (Belgium)
	Bonaventure, O. (Belgium)
	Casals, O. (Spain)
	Karlsson, G. (Sweden)
	Pavlou, G. (UK)
	Quoitin, B. (Belgium)
	Roberts, J. (France)
	Stavrakakis, I. (Greece)
	Stiller, B. (Switzerland, Germany)
	Trimintzios, P. (UK)
	Van Mieghem, P. (The Netherlands)

Authors

Al Hamra, A.
Alves, A.
Avallone, S.
Azcorra, A.
Bagnulo, M.
Barlet-Ros, P.
Biersack, E.
Blondia, C.
Boavida, F.
Bonaventure, O.
Casals, O.
Cerda, L.
Choi, D.
Curado, M.
Cushnie, J.
D'Antonio, S.
Domingo-Pascual, J.
Esposito, M.
Flegkas, P.
Fodor, V.
Garcia-Martinez, A.
Gargiulo, M.
Georgatsos, P.
Georgiadis, L.
Harju, J.
Hutchison, D.
Jacquenet, C.
Karlsson, G.
Korkmaz, T.
Koucheryavy, Y.
Krunz, M.
Kuipers, F.
Li, F.
Lopes, R.
Marsh, I.
Masip-Bruin, X.

Mauthe, A.
Moltchanov, D.
Monteiro, E.
Más Ivars, I.
Panagakis, A.
Pavlou, G.
Pescapè, A.
Popa, M.
Quadros, G.
Quoitin, Q.
Roberts, J.
Romano, S.
Serpanos, D.
Smirnov, M.
Solé-Pareta, J.
Soto, I.
Stavrakakis, I.
Stiller, B.
Swinnen, L.
Sánchez-López, S.
Tandel, S.
Traganitis, A.
Trcek, D.
Trimintzios, P.
Uhlig, S.
Urvoy-Keller, G.
Van Mieghem, P.
Van den Wijngaert, N.
Veciana, C.
Ventre, G.
Willems, G.

Sponsoring Institutions

COST TC-TIST

Table of Contents

QoS Roadmap for Future Internet Services

Michael Smirnov[1] and Jon Crowcroft[2]

[1]Fraunhofer FOKUS, Institute for Open Communication Systems,
Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany
smirnow@fokus.fraunhofer.de
http://www.fokus.fraunhofer.de/
[2]University of Cambridge, The Computer Laboratory,
William Gates Building 15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
Jon.Crowcroft@cl.cam.ac.uk
http://www.cl.cam.ac.uk/~jac22/

Abstract. This roadmap provides an outline of all the chapters of the book to help the reader find their way reading through this State of the Art report on Quality of future Internet Services. This we explain why the book is structured like this, what the logic is behind each chapter, and what the possible relations and dependencies between chapters are. At the end of the roadmap, we briefly outline the five-year history of the COST Action 263 that produced this book as its final report.

1 Introduction

Largely the term "Quality of Service" relates to the users' expections of service session quality, availability, etc. All chapters of this book either directly or indirectly answer the question: *What impact does network service quality have on the end-user perceived QoS?*

Internet traffic is complex because of unpredictable usage. Internet evolution is uncertain – its sustainable growth requires QoS for dependability and more flexibility for new services. We called this book after the Action – Quality of future Internet services: *future* is still, after nearly five years, the key operative term within this book.

The rest of this chapter is structured as follows. The next section provides outlines of all chapter topics: its content mirrors the book as a whole; however, it does not simply reproduce the introductions from each chapter: each topic is the result of the integration of work by several authors, and the chapter editors have provided their own introduction to each chapter. The next section aims at an analysis of the content of the entire book, in an attempt to outline a roadmap for future QoS services by highlighting the main *questions* and, where possible, solutions or approaches. References to particular sections are either by section number or indicated by emphasizing its **title** in bold font. The last section presents a very brief history of COST Action 263.

M. Smirnov et al. (Eds.): COST 263 Final Report, LNCS 2856, pp. 1-9, 2003.

2 Questions and Answers

Traffic Management at Large and at Small

The problem addressed by **traffic theory** (section 2.2) is: *How to model mathematically relations within the network between demand, capacity and performance?* Such a model is needed for network dimensioning - a task of assigning major capacity values to all network resources that will be used then by traffic handling mechanisms. Thus, as many studies advocate a dependency between traffic handling and traffic theory paradigms may seem natural. However, the Internet traffic handling paradigm – datagram switching – is known to be extremely difficult to capture in any traffic theory that wishes to provide pragmatic engineering methods and tools. Hence, this book advocates another traffic theory paradigm – flow awareness. Traffic theory based on flows does allow simple engineering models that do not have to take into account detailed traffic characteristics that are so difficult to obtain and predict. Consequently, flow aware QoS networking is shown to be applicable to both elastic and streaming types of traffic.

Engineering methods similar to that used in traditional telephony and based on Erlang formula can be obtained for the Internet traffic when viewed as flow process. The Internet blocking probability will clearly elucidate what guarantees are feasible and what controls (for example a threshold of obtained bandwidth for admitted flows) in traffic handling mechanisms are need to achieve those.

The flow aware traffic theory is needed not only to help in network dimensioning but also to provide insights into mechanisms for traffic handling. Several such insights showing dependencies between traffic theory and traffic control are found in the next chapter. One example is recommendation to implement elastic (e.g. HTTP over TCP) traffic controls such that performance is largely insensitive to the precise distribution of document sizes. TCP's closed loop control is dependent on past and current congestion status of a buffer, thus a correct model should be based on a number of flows contending for path's capacity. This correct model can take advantage of Poisson flow arrival process. Other insights include the advantageous combination of elastic traffic and streaming traffic admitted based on its peak rate but being a tiny fraction of elastic one, and clear motivation for a measurement based admission control. As a result, traffic theory chapter outlines flow aware QoS framework as an alternative to both IETF approaches: reservation (IntServ) and service differentiation (DiffServ).

Internet traffic can be classified very roughly as streaming or elastic. The **Multi-class best effort** paradigm (section 2.3) provides generalisation of the above dichotomy attempting to solve the following 'single metric' problem: *How to find a single QoS metric that could be used for characterisation of any application QoS need*? Datagram handling mechanisms based on this single metric would treat datagrams belonging to different service classes proportionally to the degradation of the metric. That is why degradation is a possible answer to the single metric question; it can be instantiated for delay and for packet loss and can be measured in units of congestion index. Packets from different services classes are handled within the best-effort service model enhanced by dynamic degradation distribution: for each traffic class

congestion indices are the same at any time period, though effective handling of classes is proportional to class's tolerance to the degradation.

Service models, like the multi-class best effort described above, can be equally attributed to both traffic theory (traffic management at large time scales) and to traffic handling (traffic management at short and medium time scale). The next type of traffic management is based on a snapshot of network state information and is intended only for short time scale management decisions. **Measurement admission control schemes** (section 2.4) address the following question: *How to provide per flow QoS guarantees in a radically distributed manner and without keeping any flow or aggregate state within the network?* Clearly, neither IntServ nor Diffserv, especially when coupled with contracted access (capacity admission control with corresponding management of service level agreements), can meet the above statelessness requirement. On the other hand, network over-dimensioning is stateless but does not provide guarantees, and, crucially, at times when guarantees are mostly needed. Thus, measurement based schemes remain major candidate technology for short-term traffic management decisions, though this area of research still has plenty of open issues.

Maybe, the key issue is *admission control placement* within a network: radical distribution can be achieved either by placing controls in each node (per hop admission control) or in each host (probe-based admission control, or network feedback based admission control). Observations made on several inherently different per hop measurement based admission control schemes reveal that they exhibit similar performance regardless the scheme's complexity. Even if based on minimal complexity principle, per hop control schemes require a lot of research to tune their admission policies properly. The same applies to placement of controls at endpoints (e.g. probe based admision control), however due to obeying the End-To-End principle of the Internet these schemes inherit all its advantages. For example, multicast extension comes at a very low additional price.

The admission policies mentioned above are initially specified as business goals, with service level agreements being an overall regulator of a user and a network behaviour in small (single session) and in large (VPN) scenarios. A **Component-based approach to QoS monitoring** (section 2.5) seeks to answer the question: *How a user and a network provider can continuously monitor mutual QoS contract fulfillment and re-negotiate it, when needed?* SLA is a formal description of the contracted network service; a service becomes a commodity, and as such can be evaluated against contracted values. Service evaluation requires service level monitoring that in turn requires tailored metering. Thus, a provider of contracted network services needs to automate two strongly related processes: SLA based service provisioning and SLA related service monitoring. Exactly this relation is reflected in the approach presented in this book: monitoring part is formalized by a document, called Service Level Indication linked to an SLA. It is proposed to automate creation of SLA by means of mediation – a process of multi-entity co-ordination and negotiation triggered by a service request. Tailoring of generic network instrumentation (metering infrastructure) is achieved by metering policies derived from SLA, thus SLA based network service is a primary one and SLA based monitoring service is a meta-service. To the best of our knowledge this is the first attempt to describe traffic management by means of co-operating business processes realised as mediation and policy-based monitoring.

All the problems and approaches considered previously - traffic demand/capacity modeling, traffic QoS metrics, stateless QoS guarantees, and QoS contract monitoring - can be summarized as traffic management at large: they in turn make use of traffic management at small time scales like buffer management and scheduling, often tuned for particular payload types. Efficient use of traffic management mechanisms in the small requires that their behaviour is well understood and, if possible deterministic. Section (2.6) considers **Generalized Processor Sharing (GPS) scheduling discipline** in attempt to answer the question: *How to maximize the best effort throughput deterministically while preserving guarantees of QoS sensitive traffic in GPS system?* GPS scheduling is widely used to allocate bandwidth resources to multiplexed traffic streams. The approach taken in this book considers an optimal scheduler for a mixture of QoS sensitive and best effort flows, in which optimal GPS coefficients are derived directly from flow's QoS requirements. To guarantee high resource utilization while preserving QoS requirements in a poof-based manner the section proposes scheduling based on a virtual system, where some flows in turn are decomposed into a pair of components - long-term and bursty.

MPEG traffic (section 2.7) is probably the best illustration for the above two-component representation for it has very high peak rate and drastic rate changes at short time scale. An easy-to-use and realistic model of moving picture traffic compressed by MPEG is an asset from many viewpoints. Precise modelling of MPEG source is possible however computationally difficult, thus the section proposes a two step engineering model. At the first phase video is modelled at frame level, and at the second phase – at the level of group of pictures.

As soon as we know how to model traffic, we could use these models for traffic management purposes. Section (2.8), An Open Architecture for DiffServ-enabled MPLS networks, seeks to answer the question: *Do we already have QoS aware mechanisms for efficient traffic engineering and for service provisioning at the Internet backbone level?* The answer is positive, namely – by using a combination of MPLS, DiffServ and explicit routing. MPLS, a so-called sub-IP technology, is much closer to flow-aware networking than hop-by-hop routed IP, which makes results of section 2.2 directly applicable to traffic management in production networks. Experimental results with focus on existing implementations demonstrate that, when MPLS is combined with Diffserv and explicit routing, we have a powerful architecture, which allows both traffic engineering and quality of service provisioning, including dynamic adaptation of static DiffServ configuration.

Another aspect of traffic type-specific, that of monitoring is suggested in Section (2.9) - **wide area measurements of Voice over IP quality**. The aim of this work is to answer a couple of question: *Is it feasible to build and operate Internet-wide monitoring infrastructure for VoIP quality? What are then needed metrics and methodology?* The section also analyses measured VoIP quality results over years to see whether the quality is improving over time. The truly global dimension of this study is clear from some numbers: 72 different paths were used over a 15 week period for just over 18,000 recorded sessions with nearly 33 million individual packets. The conclusion is that quality of VoIP is very good and in most cases it is over the requirements stated in many speech quality recommendations. Several sites participating in the effort were those of COST263 member organizations.

This empirical work on studying the QoS properties of Internet paths concludes the chapter on traffic management and makes a natural bridge to the next chapter that

aims to route datagrams from QoS sensitive flows automatically over network paths that satisfy application requirements.

Quality of Service Routing

The problem of QoS Routing – *How to automatically route datagrams of QoS sensitive flows over network paths that satisfy application requirements?* - is hard due to the fact that algorithms of finding such paths - Multi-Constrained Optimal Path algorithms – are known to be NP complete. The first part of the chapter provides thorough, concise and fair evaluation of major known MCOP algorithms. Not only an application QoS is a concern in this evaluation but network link utilization as well. Link metric is multi-dimensional, while QoS routing path metric is either additive (e.g. delay), or min/max (e.g. bandwidth, policy). Heuristics are compared based on their complexity to deliver an optimal solution. Performance analysis of heuristics (section 3.3) is based on simulation. An interesting phenomenon is reported from this study: the success rate of algorithms does not depend on the number of components in the QoS metric.
QoS Routing is not yet deployed; therefore we do not know *How to model the dynamics of propagation of network state changes in the large (e.g. topology changes) and in the small (e.g. link metric value changes)?* Section 3.4 examines, among others a centralized, server-based and a number of distributed approaches, including source based QoS routing and path selection under inaccurate information. General to all distributed cases question - when and how to update other nodes on metric changes – can be solved by introducing triggering policies (e.g. threshold-based, and class based).
Very reactive, i.e. fast converging change propagation scheme poses yet another question: *How to guarantee stability of QoS routing?* Section 3.6 outlines a number of solutions known from literature including routing metric smoothing and/or quantisation, and a number of load balancing schemes. The list of identified open issues in QoS routing concludes this chapter.

Traffic Engineering

Traffic engineering encompasses a set of techniques that can be used to control the flow of traffic in a data networks. The main problem addressed by this chapter can be formulated as *How to determine an optimal routing plan for a whole IP network?* Routing plan optimization requires performance evaluation of a given routing system within a given network topology and for a given network load. The idea behind **intra-domain traffic engineering** advocated in section 4.3 is based on long term traffic forecast for predictable traffic in combination with dynamic routing management for handling short and medium range changes in traffic. The former is achieved by network dimensioning, while the latter – by load balancing between domain's (ingress, egress) pairs of border routers and by dynamic resource management, e.g by balancing of load between behaviour aggregates in case of DiffServ model. Though section 4.2 has very extensive survey of related works, the proposal above is strongly related to DiffServ also in that it assumes that Traffic Engineering process is triggered

by service level agreements process, for example all predictable traffic is derived from SLA contracts. The long-term traffic engineering component works as through the co-ordination of the two schedulers: the provisioning scheduler and the forecast scheduler. As with flow-aware traffic management described above network and load models have engineering complexity due to the fact that the model's units are trunks or PHB groups.

Answering the question *How to communicate traffic engineering parameters to network elements?* in line with the DiffServ deployment philosophy, i.e. through configuration policies (e.g. provisioning period, cost function), the chapter however also advocates **policy extensions to intra-domain traffic engineering** (Section 4.6) by introducing resource provisioning policies that actually define behaviours of policy consumers. Enforcement of provisioning policies is more advanced compared to that of configuration policies: it is not a simple parameter setting but requires the invocation of some logic. This is a step towards a full potential of a policy defined as "a rule defining choice in the behaviour of a system" [M. Sloman]. Thus all provisioning logic is mapped to policy controls, and this inevitably requires co-location of policy decision and policy enforcement points.

As seen from another domain, all paths inside a domain are equal, however, when looking at how domains are interconnected we observe that some domains are 'more equal' than others. This inequality reflects business goals of domain providers and is expressed in transit policies, which in turn are implemented as Border Gateway Protocol input and output filters. Therefore **BGP-based inter-domain traffic engineering** (section 4.9) is currently the major tool for optimization of routing plan at the level of domains (Autonomous Systems). BGP is probably the main tussle space in the Internet due to conflicting goals the inter-domain routing needs to satisfy, thus we consider of paramount importance the following question: *Is it possible to optimize inter-domain route selection by manipulating BGP decision process?*

The approach taken is based on building a model of BGP behaviour and after that proposing optimization means. Together with input and output filters that reflect provider's policy BGP's decision making process can be seem as a set of rules. This rule-set works far from optimal; experimental results obtained by authors show that currently 50% of inter-domain roots are being chosen in non-deterministic manner. Hence, they serve as a source of potential optimization by influencing the BGP decision process. One way of answering the question above would be to play with BGP parameters trying to tune the path selection. This was studied by simulation; the study reported here is innovative because it comes one order of magnitude closer to the size of the entire Internet topology than any previous simulation, though still is one of order of magnitude smaller than the real Net.

The conclusion from the simulation analysis shows that the real impact of inter-domain traffic engineering cannot be achieved by just playing with BGP parameters, we need other techniques, and in particular **propagation of QoS information** discussed in Section 4.10. The approach is to define a new attribute to add value incrementally to BGP decision process. As a crucial for inter-domain QoS result careful manipulation of rules comprising the BGP decision logic becomes possible.

It would be unfair to forget another powerful but tricky method of connectivity control - **multi-homing**. Section 4.11 presents a survey of multi-homing solutions suitable for IPv4/IPv6 deployment at the level of AS, host, router, and site.

Mobile Networking

Mobility is taking over the Internet and introducing new challenges in the best effort and in the QoS aware IP worlds. One main question *How to support mobility of IP hosts?* is being split into a number of technical challenges. Chapter 5 analyzes these challenges formally and through simulation and outlines candidate solutions. The mechanism to support mobility is stateless auto-configuration, however it may cause collisions between randomly generated host identifiers. Bounds proposed in section 5.2 show that at least in near future collisions will not be a problem.
In section 5.3, protocols are investigated that aim at realizing seamless handover, i.e. handovers without packet loss and without introducing extra packet delay and delay jitter. Their performance is studied as a function of several system parameters.
The TCP transport protocol used in the Internet is a variable window protocol where losses are considered to be congestion signals. This may cause TCP to behave poorly in wireless networks, which are characterized by losses due to transmission errors and handoffs. Section 5.4 investigates the impact of transmission errors on TCP and the causes of packet losses during the handoffs.
In general mobility and wireless seem to be the hottest research field of the first decade of the 21st century, this is proved by the attention achieved by the relevant strategic objectives of the 6th European Framework Program, by an explosion of international conferences and workshops, not to mention growing user expectations for innovative applications enabled by all-IP always-on ambient connectivity.

Algorithms for Scalable Content Distribution

The problem is: *How to distribute content to large groups where members might be heterogeneous and/or mobile?* Truly scalable solution needs to capitalize on IP multicast, however because layer 3 multicast is not deployed widely a pragmatic solutions may be either above layer 3 (e.g. an overlay network with application level multicast) or below layer 3 (e.g. based on a satellite coverage). Chapter 6 is structured correspondingly.
The overlay problem *How to optimize realistically and pragmatically large scale content distribution system?* is addressed in Section 6.2 that introduces in fact a methodology for **cost-optimal dimensioning of a large scale video on demand system**. This methodology reveals itself as offering solutions for a tightly coupled set of technical problems, though carefully isolated into tractable analytically optimization tasks. Overall, this section develops an "analytical model for a video distribution architecture that allows to compute for a video with a given popularity the cost-optimal partitioning into prefix and suffix and the placement of the prefix servers in the distribution tree".
The proposed model has a remarkable number of real life parameters, and because of this the same model is applicable to full-size videos and to short clips, and to sets of videos, and to servers with limited resources. Basically, the model adapts itself! This broad usage spectrum is inherently possible due to the proposed two-tier control architecture that nicely separates central suffix server (controlled by open loop scheme) from prefix management (controlled by closed loop scheme), there is no need to convey user commands to the central server – scalability in signaling.

Solutions obtained with the model are cost efficient, section authors demonstrate that cost is optimal with total optimization; if one degree of freedom is removed the total system cost can increase significantly.
The completeness of the model and of the considered scenarios do not prevent future research in the same direction, for example study of applicability in peer-to-peer context, or in combination with satellite distribution of content.
Satellite systems are unbeatable when there is a need for a wide geographical coverage. Section 6.3 studies **scheduling objects for broadcast systems**, where the main question is *How to schedule segments of content to be delivered over satellite to hosts that are connected occasionally?* As usual for delay tolerant networking authors assume that clients may cache content – this is a non-traditional though realistic setting for a satellite-based broadcast system. Another assumption: clients do not have uplink connectivity; they join the broadcast group off-line and the system pushes the content. The model presented achieves optimal though different schedules for clients with caching capability and for cacheless clients as well.

Pricing and QoS

This chapter addresses the state of the art of pricing for Internet services and its relation to Quality of Service. Approaches to pricing - user-centered, content-based, and a cost sharing follow essential economic and technology basics, covering terminology, accounting, and security.
Technical implications from charging for QoS include the need for congestion control. The chapter serves as a good introduction into the area as well as an overview of practice (e.g. deployed charging schemes in NRNs), research (e.g. projects in the area), and needed network instrumentation (e.g. AAA and accounting platforms). Security issues are discussed with regard to cost related attributes.
Open research issues in pricing for QoS are not limited to those outlined in this chapter; virtually every topic covered in this book is in certain relation to pricing.

3 About This Book

This book is a collective effort of many people that were included in the European Action “Quality of future Internet Services” COST 263 (http://www.fokus.fraunhofer.de/cost263). The Action was inaugurated 29.01.99 in Brussels. COST263 Memorandum of Understanding declares as the main objective for the Action “to coordinate from a European perspective concerted actions among involved European participating organizations and research groups being active in the field of the Quality of Internet Services, the COST sponsorship will be also used for existing events, to give them the appropriate focus”. We see the book as a result of this coordination.
In 1999 we met three times (15-16.04.99 in Universita di Napoli Federico II, in Naples, Italy; 15-16.07.99 at IETF45 in Oslo Radisson Plaza in Oslo, Norway; 05-06.10.99 in Universitat Politecnica de Catalunya, in Barcelona, Spain) before we have launched our flagship event – International workshop on Quality of future Internet Services (QofIS). QofIS 2000 Programme Committee had a meeting during ‘Quality

of Service in Networks and Distributed Systems' seminar at Schloss Dagstuhl (08-09.05.2000, Schloss Dagstuhl, Germany). The 1st QofIS (25-27.09.00) was organized by Fraunhofer FOKUS in Berlin, Germany and gathered 160 participants.
Between workshops the Action was making its internal technical meetings (07-08.12.00, hosted by MATAV in Budapest, Hungary; 05.06.01 hosted by Karlsruhe University, Germany - a day before IWQoS 2001). The 2nd QofIS 2001 (24-26.09.01) was organized by Coimbra University in Coimbra, Portugal and was attended by 74 participants.
The decision to start writing this book was made in the Action's meeting (17-18.12.01) hosted by the University of Namur, with financial support from the Government of the Walloon region, Belgium. The structure of the book was decided; chapter editors and reviewers were selected. While main work of writing contributions to the book was coordinated by e-mail and shared workspace at BSCW server, we continued book discussions during PfHSN 2002 workshop (22-25.04.02) organized by FOKUS in Berlin, Germany. Major discussion about book structure, especially concerning traffic engineering chapter did happen during QofIS 2002 (16-18.10.02) organized by ETHZ in Zurich, Switzerland and attended by some 80 participants. At the same time it became clear that the two COST Actions – COST 263 and COST 264 'Networked Group Communication' will join forces to organize under the 6th European Framework Program a network of excellence in Emerging Networking Experiments and Technologies (E-NEXT).
It happened that the final book-editing meeting (30-31.01.03, hosted by Tampere University of Technology, Tampere, Finland) coincided with two days of the lowest wintertime temperature in Tampere. Maybe because of this we made a very important decision – to organize a book as a collective monograph, so that each chapter becomes a fully integrated text, not just a collection of individual papers. The outdoor temperature difference was 60ºC when many of the book authors met next time, at the COST-IST: NSF *NeXtworking* Workshop (23-25.06.03, organized in Chania Crete, Greece by COST263 and by the university of Athens), where the approval for thee E-NEXT proposal was confirmed.
QofIS 2003 will be in Stockholm, Sweden 01-02.10.03 hosted by KTH, and it fitting that while it is the last QofIS organized by COST263, it is the first organized by E-NEXT.

Traffic Management

Gunnar Karlsson (Ed.)[1], James Roberts (Ed.)[2], Ioannis Stavrakakis (Ed.)[3], Antonio Alves[4], Stefano Avallone[5], Fernando Boavida[6], Salvatore D'Antonio[7], Marcello Esposito[7], Viktoria Fodor[1], Mauro Gargiulo[7], Jarmo Harju[8], Yevgeni Koucheryavy[8], Fengyi Li[9], Ian Marsh[10], Ignacio Más Ivars[1], Dmitri Moltchanov[8], Edmundo Monteiro[6], Antonis Panagakis[3], Antonio Pescapè[5], Goncalo Quadros[4], Simon Pietro Romano[5], and Giorgio Ventre[5]

[1] KTH, Royal Institute of Technology, Laboratory for Communication Networks, 16440 Kista, Sweden, {gk,viktoria,nacho}@imit.kth.se
[2] France Telecom R&D, DAC/OAT, 38, rue du Général Leclerc, 92794 Issy les Moulineaux, James.Roberts@francetelecom.com
[3] University of Athens, Department of Informatics and Telecommunications, Athens, Greece, {istavrak,grad0260}@di.uoa.gr
[4] Critical Software, SA,Santa Clara, 3040-032 Coimbra, Portugal, {alves,gquadros}@dei.uc.pt
[5] Universita degli Studi di Napoli Federico II, Dipartimento di Informatica e Sistemistica, Via Claudio 21, 80125 Napoli, Italy, {stavallo,pescape,spromano,giorgio}@unina.it
[6] University of Coimbra, Laboratory of Communications and Telematics 3000-140 Coimbra, Portugal, {boavida,edmundo}@dei.uc.pt
[7] ITEM - Laboratorio Nazionale CINI per l'Informatica e la Telematica Multimediali, Via Diocleziano 328, 80125 Napoli, Italy, {salvatore.dantonio,marcello.esposito,mauro.gargiulo}@napoli.consorzio-cini.it
[8] Tampere University of Technology, Institute of Communication Engineering, P.O. Box 553, 33101 Tampere, Finland, {harju,yk,moltchan}@cs.tut.fi
[9] KTH, Royal Institute of Technology,10044 Stockholm, Sweden, d97-fli@nada.kth.se
[10] Swedish Institute of Computer Science, Box 1263, 16429 Kista, Sweden, ianm@sics.se

Abstract. Quality of service in network communication is mainly assured by traffic management consisting in a variety of protocols and mechanisms whose role is to prevent the network from being congested. Traffic management has been a very active area of research for two decades following the conception of packet-switched integrated services networks, such as the Internet. This chapter provides samples of the state of the art in traffic management. It includes contributions on traffic theory, traffic and service models, quality monitoring, as well as traffic control and measurements.

M. Smirnov et al. (Eds.): COST 263 Final Report, LNCS 2856, pp. 10–79, 2003.

1 Introduction

Quality of communication services comprises several aspects such as the availability, reliability and dependability of the services. These aspects relate to the expectation of users that services are always provided when needed. Other aspects relate to the quality of established communication sessions and pertain to delay, errors and loss in the data transfer. For instance, a user of an application such as IP telephony has the expectation that an established call allows a conversation to take place. Additionally it is not rendered useless by high delay, which make the interaction difficult, by errors that make the speech quality unintelligible, or by losses which interrupt the conversation. The session's quality depends both on the design and performance of the system running the communicating application as well as on the network.

Network quality depends on the traffic load of the links in the network. This in turn depends on the amount and characteristics of the traffic entering the network, as well as the change in characteristics due to multiplexing and the routing of the traffic. This seemingly simple problem of plumbing becomes exceedingly complex when a network, such as the Internet, is large geographically, administratively, and in user population.

Traffic in the Internet is extremely complex, resulting from a very large and perpetually changing range of communication applications. Traffic management consists in characterizing, measuring, classifying and grooming this traffic in order to create order out of this seeming chaos. A vast amount of work has been performed over the years on the various aspects of IP traffic management. Service models like Intserv and Diffserv have been proposed to allow the network to offer a set of quality of service guarantees to different kinds of applications. The traffic generated by various applications has been widely studied and the statistical properties of individual flows and traffic aggregates are beginning to be fairly well understood. New routing techniques like MPLS allow more efficient traffic engineering.

Despite these significant advances, it remains true to say that much uncertainty surrounds the future evolution of the Internet. Some argue that an alternative service model is required and propose new network architectures and additional traffic controls. Research continues on characterizing traffic and on the development of analytical modelling techniques enabling performance evaluation and effective network design. This chapter presents a sample of this research drawn from the work of COST 263 participants. The subjects addressed in the following sections are quite diverse but all relate to the highly important field of traffic management.

Traffic theory consists in describing through mathematical models the relation between demand, capacity and performance. Section 2 argues that such theory should be increasingly applied to the design of traffic management for the Internet. Distinct modeling approaches are discussed for so-called elastic traffic and streaming traffic, represented in both cases at flow level. It is demonstrated that, under certain assumptions about how the bandwidth is shared, performance is largely insensitive to detailed traffic characteristics, allowing the definition of simple traffic engineering principles. It is argued finally that existing QoS architectures do not take sufficient account of the insights provided by traffic theory and that an alternative flow-aware networking architecture is necessary.

The simplest method for applications to obtain the quality of service they need is to over-dimension the network. Despite its simplicity, this solution has several drawbacks: Its financial cost, the implicit promotion of waste, the diversity of the QoS needs of applications, and the different characteristics of the protocols. Section 3 presents a new IP service model that can support traffic classes, according to the IETF Differentiated Services Model. The model appears in a context in which one recognizes that over-dimensioning communication systems is not a good solution. It is based on the belief that it is possible to obtain good results by applying simple strategies that do not collide with classical IP operational principles and technologies but, instead, are able to make the most out of their virtues.

The architecture in the previous section relates to the treatment of traffic within the network. A vital function of traffic management is to restrict the inflow of traffic to the network by means of admission control, as pointed out in Section 2. There has been an interest during the last five year to simplify measurement-based admission controls by administering the control from the edges of the network and not in each node of the network in the traditional manner. Section 4 surveys recent proposal of endpoint admission control with an emphasis on probe-based admission control. This particular scheme is based on the transmission of test packets to estimate the quality of the path for a communication session. The session is established only if a specified quality level is reached during the test phase.

The service to a user from a public network operator will be regulated in a service-level agreement. It is then of primary importance to provide mechanisms for monitoring the performance related to an SLA. This monitoring is of interest to both users and network service providers. Section 5 proposes a monitoring framework that represents the basis for the collection and distribution of performance data. The idea behind the proposal is the definition of an information document, a Service Level Indication (SLI). Monitoring information should be provided by the network to the user application by collecting and appropriately combining performance measures into an SLI that is linked to the SLA. Monitoring might hence be a meta-service, i.e. a service for a service, since it cannot exist on its own: monitoring is linked to an existing service, for which it provides added value.

Traffic management at small time scales (of the order of packet transmission time) consists of buffer management and packet scheduling. Section 6 examines the case of a single node employing the Generalized Processor Sharing (GPS) scheduling discipline for the provision of deterministic delay guarantees to leaky bucket constrained sources. The problem of call admission control is investigated and an algorithm, which is capable of always producing the optimal weight assignment is presented. In addition, a modification of GPS which is based on the idea of decomposing original sessions into traffic components that can be more efficiently handled by GPS is introduced and a call admission control algorithm for the proposed discipline is discussed.

Effective design of traffic management relies on a sound understanding of the nature of the traffic. It is also necessary to be able to succinctly describe specific types of traffic using appropriate mathematical models. Section 7 discusses such models for variable rate video traffic using MPEG coding. A two-step approach is proposed consisting of first accounting for the correlation structure at the group of pictures (GoP)

level and then applying known intra-GoP correlation properties to adjust the size distributions of successive I, P and B frames. The resulting model accurately predicts the significant characteristics of variable rate video traffic using a succinct representation in terms of discrete batch Markovian arrival processes that can be used in both analytical evaluations and simulation studies.

The Internet has quickly evolved into a mission-critical communications infrastructure, supporting significant economic, educational and social activities. At the same time, the provision of Internet communication services has become highly competitive and end-users are increasingly demanding. Consequently, performance optimization of large scale IP networks has become an important problem. Enhancing the performance of an operational network, at both the traffic and resource levels, is a major objective of Internet traffic engineering. Traffic engineering deals essentially with the selection of optimal paths that different flows should follow in order to optimize resource utilization and satisfy each flow's requirements. The development and introduction of Multi-Protocol Label Switching (MPLS) has opened new possibilities to address some of the limitations of IP systems concerning traffic engineering. Section 8 presents an experimental approach to the study of the interoperability between Diffserv and MPLS paradigms. Few implementations currently exist of both architectures and most of them represent proprietary solutions. The contribution is therefore related both to exploiting state-of-the-art open source implementations and to evaluating the possibility of introducing new functionality.

No standardized scheme exists for monitoring the quality of voice transmissions over IP networks (VoIP), as used for telephony. It is well known that VoIP users of the Internet are susceptible and sensitive to quality changes, causing them to abandon calls when the quality deteriorates below an acceptable level. A wide area monitoring scheme, managed over multiple domains, would be beneficial to allow administrators to adjust resources for users of an Internet telephony service. Section 9 therefore describes a quality monitoring scheme and summarizes results obtained by applying the scheme. Measurements were taken continuously over a number of weeks, providing more than 24000 simulated phone calls enabling the quality of VoIP to be given in the year 2002.

2 Traffic Theory and Flow Aware Networking

We argue here that traffic theory should be increasingly used to guide the design of the future multiservice Internet. By traffic theory we mean the application of mathematical modeling to explain the traffic-performance relation linking network capacity, traffic demand and realized performance. Traffic theoretic considerations lead us to argue that an effective QoS network architecture must be flow aware.

Traffic theory is fundamental to the design of the telephone network. The traffic-performance relation here is typified by the Erlang loss formula which gives the probability of call blocking, E, when a certain volume of traffic, a, is offered to a given number of circuits, n:

$$E(n,a) = \frac{a^n/n!}{\sum_{i \leq n} a^i/i!}.$$

The formula relies essentially only on the reasonable assumption that telephone calls arrive as a stationary Poisson process. It demonstrates the remarkable fact that, given this assumption, performance depends only on a simple measure of the offered traffic, a, equal to the product of the call arrival rate and the average call duration.

We claim that it is possible to derive similar traffic-performance relations for the Internet, even if these cannot always be expressed as concisely as the Erlang formula. Deriving such relations allows us to understand what kinds of performance guarantees are feasible and what kinds of traffic control are necessary.

It has been suggested that Internet traffic is far too complicated to be modeled using the techniques developed for the telephone network or for computer systems [1]. While we must agree that the modeling tools cannot ignore real traffic characteristics and that new traffic theory does need to be developed, we seek to show here that traditional techniques and classical results do have their application and can shed light on the impact of possible networking evolutions.

Following a brief discussion on Internet traffic characteristics we outline elements of a traffic theory for the two main types of demand: streaming and elastic. We conclude with the vision of a future flow aware network architecture built on the lessons of this theory.

2.1 Statistical Characterization of Traffic

Traffic in the Internet results from the uncoordinated actions of a very large population of users and must be described in statistical terms. It is important to be able describe this traffic succinctly in a manner which is useful for network engineering.

The relative traffic proportions of TCP and UDP has varied little over at least the last five years and tend to be the same throughout the Internet. More than 90% of bytes are in TCP connections. New streaming applications are certainly gaining in popularity but the extra UDP traffic is offset by increases in regular data transfers using TCP. The applications driving these document transfers is evolving, however, with the notable impact over recent years first of the Web and of peer to peer applications.

For traffic engineering purposes it is not necessary to identify all the different applications comprising Internet traffic. It is generally sufficient to distinguish just three fundamentally different types of traffic: elastic traffic, streaming traffic and control traffic. Elastic traffic corresponds to the transfer of documents under the control of TCP and is so-named because the rate of transfer can vary in response to evolving network load. Streaming traffic results from audio and video applications which generate flows of packets having an intrinsic rate which must be preserved by limiting packet delay and loss. Control traffic derives from a variety of signaling and network control protocols. While the efficient handling of control traffic is clearly vital for the correct operation of the network, its relatively small volume makes it a somewhat minor consideration for traffic management purposes.

Observations of traffic volume on network links typically reveal intensity levels (in bits/sec) averaged over periods of 5 to 10 minutes which are relatively predictable from day to day (see Figure 2.1). It is possible to detect a busy period during which the traffic intensity is roughly constant. This suggests that Internet traffic, like telephone traffic,

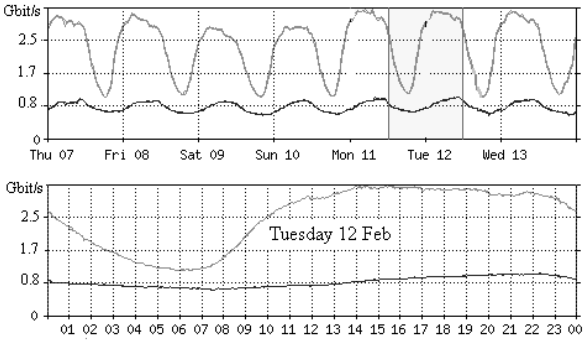


Fig. 1. Traffic on an OC192 backbone link.

can be modeled as a stationary stochastic process. Busy hour performance is evaluated through expected values pertaining to the corresponding stationary process.

The traffic process can be described in terms of the characteristics of a number of objects, including packets, bursts, flows, sessions and connections. The preferred choice for modeling purposes depends on the object to which traffic controls are applied. Conversely, in designing traffic controls it is necessary to bear in mind the facility of characterizing the implied traffic object. Traffic characterization proves most convenient at flow level.

A flow is defined for present purposes as the unidirectional succession of packets relating to one instance of an application (sometimes referred to as a microflow). For practical purposes, the packets belonging to a given flow have the same identifier (e.g., source and destination addresses and port numbers) and occur with a maximum separation of a few seconds. Packet level characteristics of elastic flows are mainly induced by the transport protocol and its interactions with the network. Streaming flows, on the other hand, have intrinsic (generally variable) rate characteristics that must be preserved as the flow traverses the network.

Flows are frequently emitted successively and in parallel in what are loosely termed sessions. A session corresponds to a continuous period of activity during which a user generates a set of elastic or streaming flows. For dial-up customers, the session can be defined to correspond to the modem connection time but, in general, a session is not materialized by any specific network control functions.

The arrival process of flows in a backbone link typically results from the superposition of a large number of independent sessions and has somewhat complex correlation behavior. However, observations confirm the predictable property that session arrivals in the busy period can be assimilated to a Poisson process.

The size of elastic flows (i.e., the size of the documents transferred) is extremely variable and has a so-called heavy-tailed distribution: most documents are small (a few kilobytes) but the number which are very long tend to contribute the majority of traffic. The precise distribution clearly depends on the underlying mix of applications (e.g.,

mail has very different characteristics to MP3 files) and is likely to change in time as network usage evolves. It is therefore highly desirable to implement traffic controls such that performance is largely insensitive to the precise document size characteristics.

The duration of streaming flows also typically has a heavy-tailed distribution. Furthermore, the packet arrival process within a variable rate streaming flow is often self-similar [2, Chapter 12]. As for elastic flows, it proves very difficult to precisely describe these characteristics. It is thus again important to design traffic controls which make performance largely insensitive to them.

2.2 Traffic Theory for Elastic Traffic

Exploiting the tolerance of document transfers to rate variations implies the use of closed-loop control to adjust the rate at which sources emit data. In this section we assume closed-loop control is applied end-to-end on a flow-by-flow basis using TCP.

TCP realizes closed loop control by implementing an additive increase, multiplicative decrease congestion avoidance algorithm: the rate increases linearly in the absence of packet loss but is halved whenever loss occurs. This behavior causes each flow to adjust its average sending rate to a value depending on the capacity and the current set of competing flows on the links of its path. Available bandwidth is shared in roughly fair proportions between all flows in progress.

A simple model of TCP results in the following well-known relationship between flow throughput B and packet loss rate p :

$$B(p) = \frac{\text{constant}}{\text{RTT}\sqrt{p}}.$$

where RTT is the flow round trip time (see [3] for a more accurate formula). This formula illustrates that, if we assume the loss rate is the same for all flows, bandwidth is shared in inverse proportion to the round trip time of the contending flows.

To estimate the loss rate one might be tempted to deduce the (self-similar, multifractal) characteristics of the packet arrival process and apply queuing theory to derive the probability of buffer overflow. This would be an error, however, since the closed-loop control of TCP makes the arrival process dependent on the current and past congestion status of the buffer. This dependence is captured in the above throughput formula.

The formula can alternatively be interpreted as relating p to the realized throughput B. Since B actually depends on the set of flows in progress (each receiving a certain share of available bandwidth), we deduce that packet scale performance is mainly determined by flow level traffic dynamics. It can, in particular, deteriorate rapidly as the number of flows sharing a link increases.

Consider the following fluid model of an isolated bottleneck link where flows arrive according to a Poisson process. Assume that all flows using the link receive an equal share of bandwidth ignoring, therefore, the impact of different round trip times. We further assume that rate shares are adjusted immediately as new flows begin and existing flows cease.

The number of flows in progress in this model is a random process which behaves like the number of customers in a so-called processor sharing queue [4]. Let the link

bandwidth be C bits/s, the flow arrival rate λ flows/s and the mean flow size σ bits. The distribution of the number of flows in progress is geometric:

$$\Pr[n \text{ flows}] = (1-\rho)\rho^n,$$

where $\rho = \lambda\sigma/C$ is the link utilization. The expected response time $R(s)$ of a flow of size s is:

$$R(s) = \frac{s}{C(1-\rho)}.$$

From the last expression we deduce that the measure of throughput $s/R(s)$ is independent of flow size and equal to $C(1-\rho)$.

It is well known that the above results are true for any flow size distribution. It is further shown in [5] that they also apply with the following relaxation of the Poisson flow arrivals assumption: sessions arrive as a Poisson process and generate a finite succession of flows interspersed by think times; the number of flows in the session, flow sizes and think times are generally distributed and can be correlated. Statistical bandwidth sharing performance thus depends essentially only on link capacity and offered traffic.

Notice that the above model predicts excellent performance for a high capacity link with utilization not too close to 100%. In practice, flows handled by such links are limited in rate elsewhere (e.g. in the access network, by a modem). The backbone link is then practically transparent with respect to perceived performance.

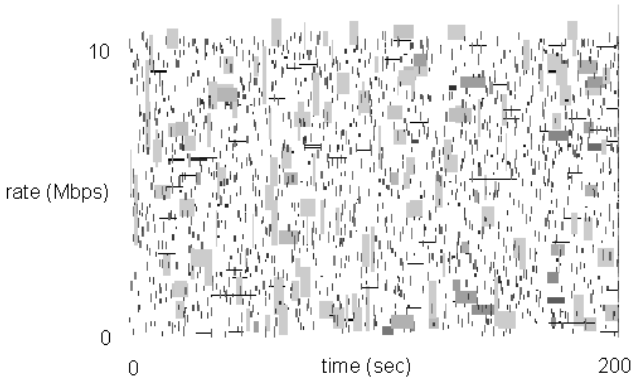


Fig. 2. Depiction of flow throughput for 90% link load.

The above model of ideally fair bandwidth sharing usefully illustrates two interesting points which turn out to be more generally true. First, performance depends primarily on expected traffic demand (in bits/second) and only marginally on parameters describing the precise traffic process (distributions, correlation). Second, backbone performance tends to be excellent as long as expected demand is somewhat less than available capacity. The latter point is illustrated in Figure 2.

The figure depicts the throughput of flows traversing a bottleneck link of capacity 10 Mbps under a load of 90% (i.e., flow arrival rate $\times$ average flow size = 9 Mbps), as evaluated by ns2 simulations. Flows are represented as rectangles whose left and right coordinates correspond to the flow start and stop time and whose height represents the average throughput. The flow throughput is also represented by the shade of the rectangle: the lightest grey corresponds to an average rate greater than 400 Kbps, black corresponds to less than 20 Kbps. In Figure 2, despite the relatively high load of 90%, most flows attain a high rate, as the model predicts.

In overload, when expected demand exceeds link capacity, the processor sharing queue is unstable: the number of flows in progress increases indefinitely as flows take longer and longer to complete while new flows continue to arrive. Figure 3 shows a rectangle plot similar to Figure 2 for an offered load equal to 140% of link capacity. The rectangles tend to become black lines since the number of competing flows increases steadily as the simulation progresses.

In practice, instability is controlled by users abandoning transfers, interrupting sessions or simply choosing not to use the network at all in busy periods. The end result is that link capacity is inefficiently used while perceived throughput performance becomes unacceptable, especially for long transfers [6]. An effective overload control would be to implement some form of proactive admission control: a new flow would be rejected whenever the bandwidth it would receive falls below a certain threshold. Figure 4 is the rectangle plot of the bottleneck link under 140% load with the application of admission control. Flow throughput is maintained at around 100 Kbps represented by the medium grey colour of the rectangles.

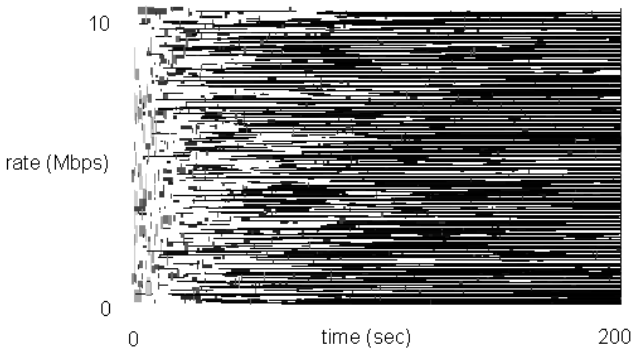


Fig. 3. Depiction of flow throughput for 140% link load.

The above traffic theory does not lead to an explicit traffic-performance relation showing how a provider can meet precise throughput guarantees. In fact, consideration of the statistical nature of traffic, the fairness bias due to different round trip times and the impact of slow-start on the throughput of short flows suggests that such guarantees are unrealizable. A more reasonable objective for the provider would be to ensure a link

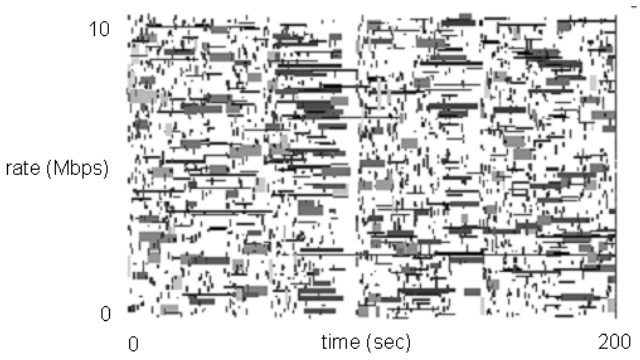


Fig. 4. Depiction of flow throughput with admission control.

is capable of meeting a minimal throughput objective for a hypothetical very large flow. This throughput is equal to $C(1-\rho)$, even when sharing is not perfectly fair.

2.3 Traffic Theory for Streaming Traffic

We assume that streaming traffic is subject to open-loop control: an arriving flow is assumed to have certain traffic characteristics; the network performs admission control, only accepting the flow if quality of service can be maintained; admitted flows are policed to ensure their traffic characteristics are indeed as assumed.

The effectiveness of open-loop control depends on how accurately performance can be predicted given the characteristics of audio and video flows. To discuss multiplexing options we first make the simplifying assumption that flows have unambiguously defined rates like fluids. It is useful then to distinguish two forms of statistical multiplexing: bufferless multiplexing and buffered multiplexing.

In the fluid model, statistical multiplexing is possible without buffering if the combined input rate is maintained below link capacity. As all excess traffic is lost, the overall loss rate is simply the ratio of expected excess traffic to expected offered traffic, i.e., $\mathrm{E}[(\Lambda_t - C)^+]/\mathrm{E}[\Lambda_t]$ where Λ_t is the input rate process and C is the link capacity. It is important to notice that this loss rate only depends on the stationary distribution of the combined input rate but not on its time dependent properties.

The level of link utilization compatible with a given loss rate can be increased by providing a buffer to absorb some of the input rate excess. However, the loss rate realized with a given buffer size and link capacity then depends in a complicated way on the nature of the offered traffic. In particular, loss and delay performance turn out to be very difficult to predict when the input process is self-similar (see [2, p. 540]). This is a significant observation in that it implies buffered multiplexing leads to extreme difficulty in controlling quality of service. Even though some applications may be tolerant to quite long packet delays it does not appear feasible to exploit this tolerance. Simple errors may lead to delays that are more than twice the objective or buffer overflow rates that are ten times the acceptable loss rate.

An alternative to meeting QoS requirements by controlled statistical multiplexing is to guarantee deterministic delay bounds for flows whose rate is controlled at the network ingress. Network provisioning and resource allocation then rely on results from the so-called network calculus [7]. The disadvantage with this approach is that it typically leads to considerable overprovisioning since the bounds are only ever attained in unreasonably pessimistic worst-case traffic configurations. Actual delays can be orders of magnitude smaller.

Bufferless statistical multiplexing has clear advantages with respect to the facility with which quality of service can be controlled. It is also efficient when the peak rate of an individual flow is small compared to the link rate because high utilization is compatible with negligible loss.

Packet queuing occurs even with so-called bufferless multiplexing due to the coincidence of arrivals from independent inputs. While we assumed above that rates were well defined, it is necessary in practice to account for the fact that packets in any flow arrive in bursts and packets from different flows arrive asynchronously. Fortunately, it turns out that the accumulation of jitter does not constitute a serious problem, as long as flows are correctly spaced at the network ingress [8].

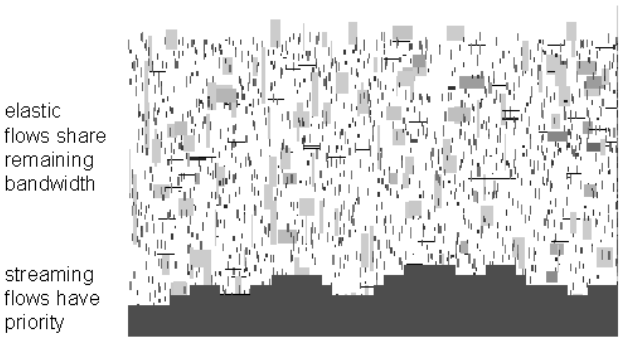


Fig. 5. Bandwidth sharing between streaming and elastic flows.

Though we have discussed traffic theory for elastic and streaming traffic separately, integration of both types of flow on the same links has considerable advantages. By giving priority to streaming flows, they effectively see a link with very low utilization yielding extremely low packet loss and delay. Elastic flows naturally benefit from the bandwidth which would be unused if dedicated bandwidth were reserved for streaming traffic and thus gain greater throughput. This kind of sharing is depicted in Figure 5.

It is generally accepted that admission control must be employed for streaming flows to guarantee their low packet loss and delay requirements. Among the large number of schemes which have been proposed in the literature, our preference is clearly for a form of measurement-based control where the only traffic descriptor is the flow peak rate and the available rate is estimated in real time. A particularly simple scheme

is proposed by Gibbens et al. [9]. In an integrated network with a majority of elastic traffic, it may not even be necessary to explicitly monitor the level of streaming traffic.

2.4 A Flow Aware Traffic Management Framework

The above considerations on traffic theory for elastic and streaming flows lead us to question the effectiveness of the classical QoS solutions of resource reservation and class of service differentiation [10]. In this section we outline a possible alternative, a flow aware network architecture.

It appears necessary to distinguish two classes of service, namely streaming and elastic. Streaming packets must be given priority in order to avoid undue delay and loss. Bufferless multiplexing must be used to allow controlled performance of streaming flows. Packet delay and loss are then as small as they can be providing the best quality of service for all applications. Bufferless multiplexing is particularly efficient under the realistic conditions that the flow peak rate is a small fraction of the link capacity and the majority of traffic using the link is elastic. Elastic flows are assumed to fairly share the residual bandwidth left by the priority streaming flows. From results of the processor sharing model introduced above, we deduce that the network will be virtually transparent to flow throughput as long as the overall link load is not too close to one. This observation has been confirmed by NS simulations of an integrated system [11]. Per flow admission control is necessary to preserve performance in case demand exceeds link capacity. We advocate applying admission control similarly to both streaming and elastic flows. If elastic traffic is the majority (at least 50%), the admission decision could be based simply on a measure of the bandwidth currently available to a new elastic flow. If the relative streaming traffic volume increases, an additional criterion using an estimation of the current overall streaming traffic rate could be applied as proposed in [9].

Implementation of flow aware networking obviously requires a reliable means of identifying individual flows. A flow identifier could be derived from the usual microflow 5-tuple of IPv4. A more flexible solution would be to use the flow label field of the IPv6 packet header allowing the user to freely define what he considers to constitute a 'flow' (e.g., all the elements of a given Web page). Flow identification for admission control would be performed 'on the fly' by comparing the packet flow identifier to a list of flows in progress on a controlled link. If the packet corresponds to a new flow, and the admission criteria are not satisfied, the packet would be discarded. This is a congestion signal to be interpreted by the user, as in the probe-based admission schemes discussed in Section 4. If admissible, the new flow identifier is added to the list. It is purged from the list if no packet is observed in a certain interval.

Admission control preserves the efficiency of links whose demand exceeds capacity. Rather than rejecting excess flows, a more satisfactory solution would be to choose an alternative route. This constitutes a form of adaptive routing and would considerably improve network robustness and efficiency compared to that currently offered by Internet routing protocols.

Admission control can be applied selectively depending on a class of service associated with the flow. Regular flows would be rejected at the onset of congestion and

then premium flows if congestion degrades further. This constitutes a form of service differentiation with respect to accessibility.

Note finally that all admitted flows have adequate quality of service and are therefore subject to charging. A simple charging scheme is appropriate based on byte counting without any need to distinguish different service classes: streaming flows experience negligible packet loss and delay while elastic flows are guaranteed a higher overall throughput.

3 An IP Service Model for the Support of Traffic Classes

3.1 Introduction

The project that led to the development of the IP service model presented in this text started with the development of a metric for evaluating the quality of service in packet switched networks. Such a metric, presented in [12] and hereafter named QoS metric, is aimed at measuring quantifiable QoS characteristics [13] in communication systems, such as throughput, transit delay or packets loss. It is especially tailored to the intermediary layers of communication systems, namely the network layer. During the metric development some ideas arose and were subsequently refined, progressively leading to a novel IP service model.

The central idea behind the proposed model is still to treat the traffic using the classical best effort approach, but with the traffic divided into several classes instead of a single class. This corresponds to a shift from a single-class best-effort paradigm to a multiple-class best-effort paradigm. In order to do this, a strategy which dynamically redistributes the communication resources is adopted, allowing classes for which degradation does not cause a significant impact to absorb the major part of congestion, thereby relieving the remaining classes.

Classes are characterized by traffic volumes, which can be potentially very different, and can be treated better or worse inside the communication system according to their needs and to the available resources. This means that classes may suffer better or worse loss levels inside the network and their packets may experience larger or smaller transit delays. The model's capacity to differentiate traffic is achieved by controlling the transit delay and losses suffered by packets belonging to different classes, through the dynamic distribution of processing and memory resources. The bandwidth used by the traffic is not directly controlled, because the characteristics of the model make it unnecessary.

3.2 Service Model

The model proposal discussed here follows the *differentiated services* architecture [14] and considers that traffic is classified into classes according to its QoS needs. The central idea is still to treat the traffic using the *best effort* approach, but with the traffic divided into several classes instead of a single one. Thus, traffic is still treated *as good as possible*, which has different implications for each considered class.

In order to implement this idea, a strategy is adopted which dynamically redistributes the communications resources, allowing classes for which degradation does

not have a significant impact to absorb the resource deficiency, thereby relieving the remaining ones.

The model's capacity to differentiate traffic is achieved by controlling the transit delay and losses suffered by packets of the different classes (which is the goal of the dynamic distribution of resources). The bandwidth used by the traffic is not controlled. Given the characteristics of the model, this kind of control is not necessary as well as not being possible.

In fact, as one of the main goals of the model is to avoid complexity, traffic specifications and explicit resource reservations are not considered. As a result, it is not possible to anticipate the bandwidth used by the different classes and, consequently, it does not make sense to base a strategy of differentiation on the active control of this QoS characteristic. Classes are characterized by very different volumes, which can be treated better or worse inside the communication system. This means that classes may suffer better or worse loss levels inside the network and their packets may experience higher or lower transit delays. Consequently, this model exerts its action controlling these two characteristics.

The proposed IP service model is based on the following three main components:

- network elements, comprising the resources and mechanisms that implement the *multiple-class-best-effort* service;
- dynamic QoS-aware routing, which accounts for the routing of traffic taking into account its QoS needs;
- communications system management, which accounts for the dimensioning and operability of the communication system, including traffic admission control functions, the main goal of which is to avoid scenarios of extreme high load.

The model presented here proposes the evolution of the *single-class-best-effort* paradigm, currently used in the Internet, into a *multiple-class-best-effort* paradigm. One of the most important challenges for building such a paradigm, given that the framework of this proposal is the IETF DS model, is the definition of a PHB able to control the behavior of network elements, named D3 PHB.

In general terms, traffic is divided into classes according to their sensitivity to transit delay and packets loss degradation. As a result of network elements behavior, traffic with higher sensitivity to degradation is protected at the expense of less sensitive traffic. Thus, the idea is to dynamically and asymmetrically redistribute the degradation among the different classes, protecting some classes at the expense of others – hence the name D3, which stands for *Dynamic Degradation Distribution*.

The strategy for this effect is built on measuring continuously the quality of service given to each traffic class and, according to the obtained measures, adjusting the mechanisms responsible for packet processing dynamically. The question is which criterion are reasonable or sensible to be used for a degradation distribution among classes that is considered?

The metric's main principle is to evaluate the impact of the variations of QoS characteristics (not the QoS characteristics variations themselves) in one or the other direction with respect to the normal range of values. The metric defines degradations and excess zones which, in turn, define how the impact varies with QoS characteristics variations.

It defines such zones for each QoS characteristic, e.g. transit delay or packets loss, and for each traffic flow or traffic class. Through the aggregation of finer measurements, e.g. the ones corresponding to a given QoS characteristic of a given flow, it is possible to construct broader measures, e.g. measurements corresponding to a given traffic class, constituted by a combination of a given set of QoS characteristics.

The D3 PHB was first presented in [15], from which Figure 6 has been extracted. In this figure (upper left corner), an example is presented of three classes with different sensitivity to transit delay degradation – high, medium and low, respectively. The network elements' main goal is to guarantee that the impact of the transit delay and packet loss degradation on the applications is the same for all the three classes to facilitate the presentation let us consider for now transit delay only. Therefore, network elements must control the transit delay suffered by packets of the three different classes in such a way that the congestion indexes related to this QoS characteristic are the same for all classes. Considering again Figure 6 as a reference, suppose that for a certain load level, which happens in the instant of time t1, this impact is evaluated by the index value CI_{t1}. Then, the transit delays suffered by the packets of each class are, from the most to the least sensitive, d1, d2 and d3, respectively.

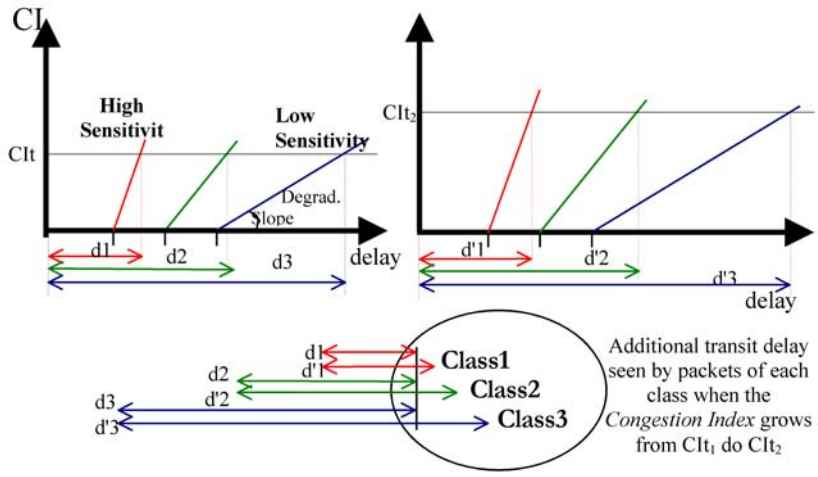


Fig. 6. Congestion indexes for three traffic classes with different sensitivity to delay degradation.

In a more formal manner, the above mentioned goal means that the following equation holds for any time interval $[\mathrm{t}_i\ ,\ \mathrm{t}_{i+1}]$:

$$\begin{aligned}\mathrm{CI}_{\mathrm{avg}\,Delay}^{Class\_1}[\mathrm{t_i}, \mathrm{t_{i+1}}] = \mathrm{CI}_{\mathrm{avg}\,Delay}^{Class\_2}[\mathrm{t_i}, \mathrm{t_{i+1}}] = ... = \\ \mathrm{CI}_{\mathrm{avg}\,Delay}^{Class\_n}[\mathrm{t_i}, \mathrm{t_{i+1}}] = \mathrm{CI}_{\mathrm{avg}\,Delay}[\mathrm{t_i}, \mathrm{t_{i+1}}]\ .\end{aligned} \tag{1}$$

Exactly the same reasoning should be made if packet losses, instead of transit delay, are used as the QoS characteristic for evaluating the degradation of quality of service. In this case the formula which applies to any time interval $[t_i , t_{i+1}]$ is the following:

$$\begin{aligned} \mathrm{CI}_{\mathrm{avg}\,Loss}^{Class\_1}[t_i, t_{i+1}] = \mathrm{CI}_{\mathrm{avg}\,Loss}^{Class\_2}[t_i, t_{i+1}] = ... = \\ \mathrm{CI}_{\mathrm{avg}\,Loss}^{Class\_n}[t_i, t_{i+1}] = \mathrm{CI}_{\mathrm{avg}\,Loss}[t_i, t_{i+1}] \ . \end{aligned} \quad (2)$$

Thus, equations 1 and 2 establish the formal criterion which governs the network elements' behavior in respect to the way IP packets are treated. Through its simultaneous application, network elements control, in an integrated way, the transit delay and loss level suffered by the different classes.

To understand the way traffic from classes could be more sensitive to degradation. , let us consider again Figure 7. Its upper left corner represents a load situation that corresponds, as seen before, to a congestion index equal to CI_{t1}.

Suppose that at a given instant of time, t2, the load level to which the network element is submitted, rises. The impact of the degradation experienced by the different applications, which generate traffic for the different classes, will also rise. The mechanisms of the network element will adjust themselves taking into account the criterion for resource distribution. The values of such an impact, the congestion indexes, must be equal for all the classes. As can be seen in Figure 7 (right side), this corresponds to transit delays from the most to the least sensitive class of d'1, d'2 and d'3 respectively.

It is possible to see clearly in the figure that the value of (d'1-d1) is lower than the value of (d'2-d2) which, in turn, is lower than the value of (d'3-d3). Thus, the increase in transit delay suffered by packets of the different classes, when the load grows, is lower for classes that are more sensitive to transit delay degradation and greater for less sensitive classes. Hence, the degradation that occurred at t2 was asymmetrically distributed among the different classes, the major part of it being absorbed by the ones less sensitive to degradation. Summing up, more sensitive classes are in fact protected at the expense of less sensitive classes, which is one of the most important goals of this proposal.

Finally, it is important to say that the control of the way in which some classes are protected at the expense of others, or even of the degradation part effectively absorbed by less sensitive classes, is materialized through the definition of the degradation sensitivity for each class.

3.3 An Implementation of the D3 Per Hop Behavior

It order to verify the ideas put forward in the proposed IP service model, it was decided to build a prototype of the D3 PHB. This section describes this prototype.

The Packet Scheduler The basic idea of the work described in this section was the integration of the characteristics of the *work-conserving* (WC) and *non-work-conserving* (NWC) disciplines in one single mechanism. This was done in order to obtain a new packet scheduler, which was simple but functional and able to effectively overcome the difficulties revealed by the experiments referred to in the previous subsection.

Such a scheduler was first described in [16]. Figure 7 – extracted from that paper – presents the logical diagram of the scheduler. The figure shows the scheduler organized into two modules that reflect the two stages of the controller's development.

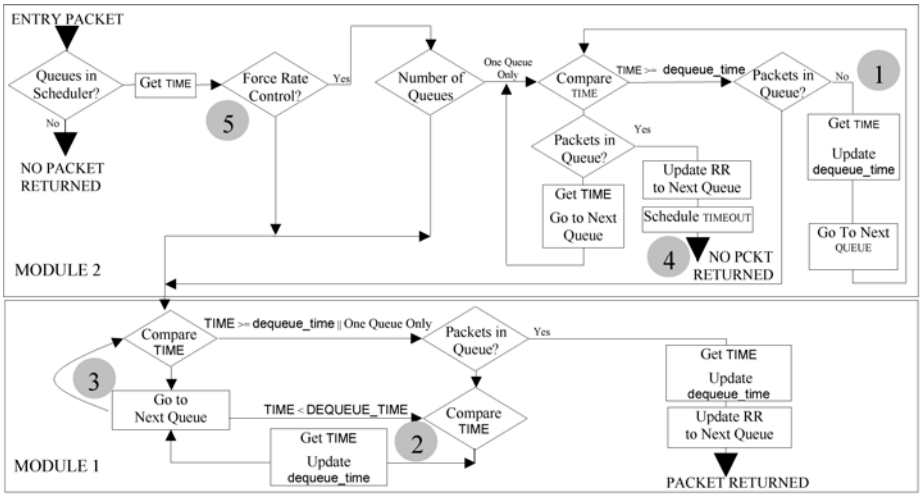


Fig. 7. Logical diagram of the scheduler implemented at LCT.

Taking DEQUEUE_TIME as the instant of time after which the controller may process the next packet in a certain queue, XDELAY as the period of time that must pass between the processing of two consecutive packets in a certain queue (that is to say, the minimum time that a packet must wait in its queue), and TEMPO as the current system time, this is, concisely, how the scheduler works:

- it sequentially visits (*round-robin*) each IP queue;
- on each visit it compares TEMPO against the value of the variable DEQUEUE_TIME that characterizes the queue; if the first is greater than the second, the packet at the head of the queue is processed;
- on each visit it updates, if necessary, the queue's DEQUEUE_TIME (which is done by adding the value of the variable XDELAY[11] to TEMPO);
- for the most important class (*the reference class*), XDELAY is zero, which means that the scheduler behaves as a work conserving scheduler. For the other classes, XDELAY will be greater than zero and will reflect the relative importance of each class. In this way, in these cases, the controller behaves as a non-work conserving scheduler.

The Packet Dropper Having developed the packet scheduler, the next challenge became the conception of an integrated solution to control the loss of packets. In this text

[11] X_DELAY means the time, in μs, that the packet must wait in its queue.

it has been referred to as the *packet dropper* (as opposed to the *packet scheduler*). In reality, the construction of the model demands far more that one packet dropper. It demands an active queue management strategy that allows not only an intelligent drop of the packets to be eliminated, but also an effective control of the loss level suffered by the different classes. Without this, the operational principle of equality of the congestion indexes related to losses cannot be accomplished.

The queue management system was first presented in [17]. In general terms, the present system involves the storing of packets in queues and their discarding when necessary. It is composed of two essential modules – the *queue length management module* (QLMM) and the *packet drop management module* (PDMM).The network element's general architecture, including the queue management system and the packet scheduler, is shown in Figure 8.

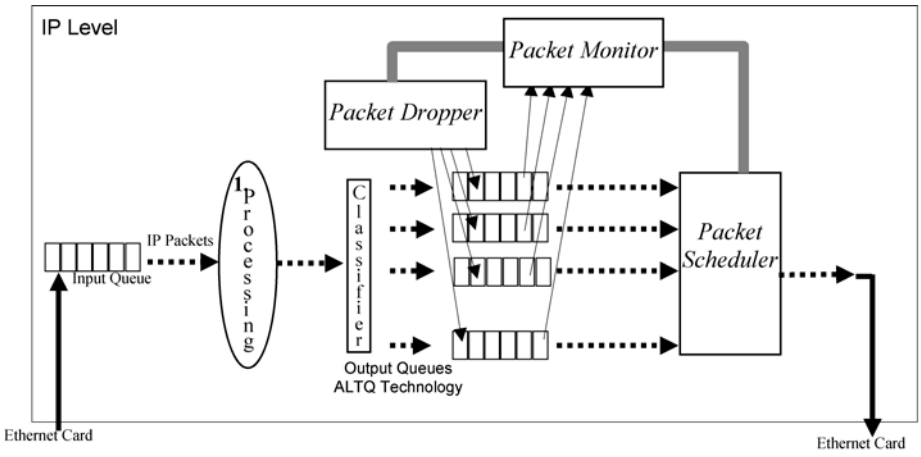


Fig. 8. Network element's general architecture [15].

3.4 Tests Made on the D3 PHB Implementation

The tests made on the D3 implementation can be subdivided into three main groups: robustness tests, performance tests and functional tests. During the development of the work presented here, tests pertaining to each of these groups were carried out.

The tests were carried out using both the QoStat tool [18] and netiQ's *Chariot* tool [19]. QoStat is a graphical user interface tool with the following characteristics:

– Graphical and numerical real-time visualization of all values pertaining to the provision of quality of service by network elements, such as transit delay, number of processed packets, number of dropped packets per unit of time, queue lengths, and used bandwidth.

– On the fly modification of all QoS-related operational parameters of network elements, such as maximum queue lengths, WFQ/ALTQ queue weights, virtual and physical queue limits, sensitivity of classes to delay degradation and sensitivity of classes to loss degradation.

The tests made to the packet scheduler whose objective was to perform a first evaluation of the differentiation capability of the prototype, clearly showed the effectiveness of the scheduler. The tests' description and their results are presented in [16].

One example is the test carried out over a small isolated network composed of two source hosts independently connected through a router (with the new scheduler installed) to a destination host. All the hosts were connected through 100 Mbps Fast Ethernet interfaces and were configured with queues with a maximum length of 50 packets. Two independent UDP flows composed of 1400 bytes packets were generated at the maximum possible rate, using the two different source hosts. They were associated to two different classes.

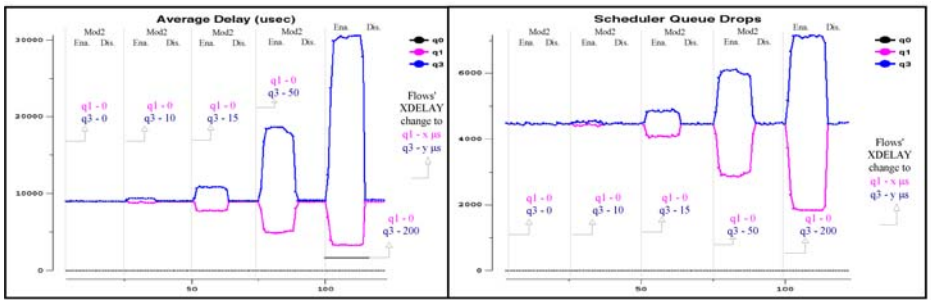


Fig. 9. left) Average IP transit delay (over 1-sec intervals); right) Number of packets sent per second

The general test strategy was the following: to set the X_DELAY associated with class 1[12] to zero, to vary the X_DELAY associated with class 3 (*qostat* tool was used to dynamically change those values, at the end of each 25 seconds time interval[13], from 0, to 10, 15, 50 and 200). Moreover, each 25 sec interval was divided into two parts. During the first part the scheduler module 2 (see Figure 7)[14] was activated; in the second part module 2 was deactivated.

Figure 9 shows the results of such a test. The left figure shows the average transit delay of packets at the IP level, measured over 1-second time intervals. The right figure the number of packets sent by the output interface per 1-second interval. It is apparent that for the referred conditions the scheduler is able to differentiate traffic. It is also

[12] Whose queue is named q1 in the figures.

[13] We did not use class 2 in the tests. As we are only using traffic of classes 1 and 3 the measured values of class 0 are always null.

[14] This module guarantees that no packet can be processed, under any circumstances, before its DEQUEUE_TIME has come, even if there is only one class with outstanding packets.

evident the importance of the scheduler module 2. Without the action of that module the scheduler is not able to differentiate traffic. Tests made to the packet dropper, or, more precisely, on the active queue management system, were carried out in two distinct phases. In the first phase, the queue length management module (QLMM) was tested. The second phase tested the packet dropper management module (PDMM). These tests are presented in [15, 17] and they will be summarized here.

Reference [16] presents results that pertain to a scenario where the sensitivity to transit delay degradation is kept constant and the sensitivity to loss degradation varies over time. A variety of tests were performed. The results of one of these tests (using two classes) are shown in Figure 10, fixed DSLOPE$_{losses}$[15], equal to 20^o for one of the flows and to 70^o for the other.

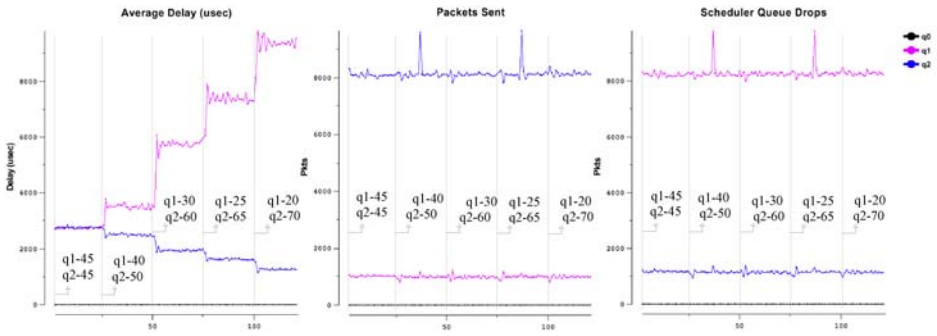


Fig. 10. Tests results in the following scenario: fixed DSLOPElosses, equal to 20^o for the q2 flow and to 70^o for the q1 flow; delay sensitivity varied over time.

It is possible to observe that, under these conditions, the behavior is as expected. The number of packets that get processed per unit of time, which is the inverse of the number of dropped packets per unit of time, is constant and consistent with the respective loss class sensitivity, DSLOPE$_{losses}$. This happened in spite of the fact that transit delay was varied over time.

3.5 Conclusions and Further Work

One of the main objectives of the work presented in this chapter was the conception of an IP service model based on the so called multiple-class-best-effort (mc-be) paradigm. This model is intended to deal with traffic classes taking into account their specific QoS needs, while still treating them *as good as possible*.

The applications do not have to specify the characteristics of the traffic they generate, nor explicitly request the QoS levels they need. Under heavy load conditions in the communication system (i.e. when performance degradation is likely to happen), they receive a better or worse treatment according to the traffic class they have chosen. This

[15] DSLOPE (*Degradation Slope*)

reflects the sensitivity to degradation preconfigured for each class, namely, the traffic sensitivity to delay degradation and the traffic sensitivity to the degradation of the level of packet losses.

The work presented here has strategically focused on the network elements. This was achieved through the conception of a Per Hop Behavior (PHB) capable of supporting the referred to model (the D3 PHB – *Dynamic Degradation Distribution*), and by the construction of the respective prototype.

The tests carried out on the prototype showed a very promising behavior with regard to its capacity for differentiating the treatment applied to the classes in a coherent and controllable way. However, in order to characterize and evaluate the presented model in a more precise manner, it is clear that the prototype must be further tested, using network load conditions closer to those which actually happen inside IP networks. The development of other components of the model also fits into future work, namely, the development of the QoS-based routing component and the traffic admission control component which, as referred to above, are currently being developed at the LCT-UC [20, 21, 22].

Meanwhile, there is a set of contributions that stand out, as results of the work presented in this text: (1) a metric for the evaluation of QoS in IP networks; (2) an IP service model that supports the mc-be paradigm; (3) a PHB that supports the model, specifically considering the network element – the D3 PHB; (4) an implementation of such a PHB model involving the building of a packet scheduler and a queuing management system.

4 Probe-Based Admission Control in IP Networks

4.1 Introduction

Today's new applications on the Internet require a better and more predictable service quality than what is possible with the available best–effort service. Audio-visual applications can handle limited packet loss and delay variation without affecting the perceived quality. Interactive communication in addition requires stringent delay requirements. For example, IP telephony requires roughly speaking a maximum of 150 ms one–way delay that needs to be maintained during the whole call.

The question of whether to provide the required service quality by over–provisioning network resources, or by admission control and reservation schemes, has been discussed extensively in the last years. In [23], Breslau and Shenker compare network performance and cost with over–provisioning and reservation. Considering non–elastic applications, their analysis shows that the amount of incremental capacity needed to obtain the same performance with a best–effort network as with a reservation–capable network diverges as capacity increases. Reservation retains significant advantages in some cases over over–provisioning, no matter how inexpensive the capacity becomes. Consequently, efficient reservation schemes can play an important role in the future Internet.

The IETF has proposed two different approaches to provide quality of service guarantees: Integrated Services (IntServ) [24] and Differentiated Services (DiffServ) [25].

IntServ provides three classes of service to the users: The guaranteed service (GS) offers transmission without packet loss and bounded end-to-end delays by assuring a fixed amount of capacity for the traffic flows [26]; the controlled load service (CLS) provides a service similar to a best–effort service in a lightly loaded network by preventing network congestion [27]; and, finally, the best–effort service lacks any kind of QoS assurances.

In the IntServ architecture, GS and CLS flows have to request admission from the network using the resource reservation protocol RSVP [28]. RSVP provides unidirectional per–flow resource reservations. When a sender wants to start a new flow, it sends a *path* message to the receiver. The message traverses all the routers in the path to the receiver, which replies with a *resv* message indicating the resources needed at every hop. This *resv* message can be denied by any router in the path, depending on the availability of resources. When the sender receives the *resv* message, the network has reserved the required resources along the transmission path and the flow is admitted. IntServ routers thus need to keep per–flow states and must process per–flow reservation requests, which can create an unmanageable processing load in the case of many simultaneous flows. Consequently, the IntServ architecture provides excellent quality in the GS class, and tight performance bounds in the CLS class, but has known scalability limitations.

The second approach for providing QoS in the Internet, the DiffServ architecture, puts much less burden on the routers, thus providing much better scalability. DiffServ uses an approach referred to as class of service (CoS), by mapping multiple flows into two default classes. Applications or ingress nodes mark packets with a DiffServ code point (DSCP) according to their QoS requirements. This DSCP is then mapped into different per–hop behaviors (PHB) at each router on the path, like expedited forwarding [29], or assured forwarding [30]. The routers additionally provide a set of priority classes with associated queues and scheduling mechanisms, and they schedule packets based on the per–hop behavior.

The drawback of the DiffServ scheme is that as it does not contain admission control. The service classes may be overloaded and all the flows belonging to that class may suffer increased packet loss. To handle overload situations, DiffServ relies on service level agreements (SLA) between DiffServ domains, which establish the policy criteria, and define the traffic profiles. Traffic is policed and smoothed at ingress points according to the SLA. Traffic that is out of profile (i.e. above the upper bounds of capacity usage stated in the SLA) at an ingress point has no guarantees and can be either dropped, over charged, or downgraded to a lower QoS class. Compared to the IntServ solution, DiffServ improves scalability at the cost of a less predictable service to user flows. Moreover, DiffServ eliminates the possibility to change the service requirements dynamically by the end user, since it would require signing a new SLA. Thus providing of quality of service is almost static.

Both IETF schemes provide end–to–end QoS with different approaches and thus with different advantages and drawbacks. Recent efforts focus on combining both schemes, like RSVP aggregation [31], the RSVP DCLASS object [32], or the proposal of the integrated services over specific link layer working group (ISSLL) to provide IntServ over DiffServ networks [33], that builds on RSVP as signaling protocol but uses DiffServ to actually share the resources among the flows.

4.2 Per-Hop Measurement Based Admission Control Schemes

Recently, a set of measurement–based admission control schemes has appeared in the literature. These schemes follow the ideas of IntServ, with connection admission control algorithms to limit network load, but without the need of per–flow states and exact traffic descriptors. They use some worst–case traffic descriptor, like the peak rate, to describe flows trying to enter the network, and then to base the acceptance decision in each hop on real–time measurements of the individual or aggregate flows.

All these algorithms focus on provisioning resources at a single network node and follow some admission policy, like complete partitioning or complete sharing. The complete partitioning scheme assumes a fixed partition of the link capacity for the different classes of connections. Each partition corresponds to a range of declared peak rates, and the partitions cover together the full range of allowed peak rates without overlap. A new flow is admitted only if there is enough capacity in its class partition. This provides a fair distribution of the blocking probability amongst the different traffic classes, but it risks lowering the total throughput if some classes are lightly loaded while others are overloaded. The complete sharing scheme, on the contrary, makes no difference among flows. A new flow is admitted if there is capacity for it, which may lead to a dominance of flows with smaller peak rate. To perform the actual admission control, measurement–based schemes use RSVP signaling.

The idea of measurement based admission control is further simplified in [34]. In this proposal the edge routers decide about the admission of a new flow. Edge routers passively monitor the aggregate traffic on transmission paths, and accept new flows based on these measurements.

An overview of several MBAC schemes is presented in [35]. This overview reveals that all the considered algorithms have similar performance, independently of their algorithmic complexity. While measurement–based admission control schemes require limited capabilities from the routers and source nodes, compared to traditional admission control or reservation schemes, like RSVP, they show a set of drawbacks: Not all proposed algorithms can select the target loss rate freely, flows with longer transmission paths experience higher blocking probabilities than flows with short paths, and flows with low capacity requirements are favored over those with high capacity needs.

4.3 Endpoint Admission Control Schemes

In the recent years a new family of admission control solutions has been proposed to provide admission control for controlled–load like services, with very little or no support from routers. These proposals share the common idea of endpoint admission control: A host sends probe packets before starting a new session and decides about the flow admission based on statistics of probe packet loss [36, 37], explicit congestion notification (ECN) marks [38, 39, 40], delay or delay variation [41, 42, 43]. The admission decision is thus moved to the edge nodes, and it is made for the entire path from the source to the destination, rather than per–hop. Consequently, the service class does not require explicit support from the routers, other than one of the various scheduling mechanisms supplied by DiffServ, and possibly the capability of marking packets.

In most of the schemes the accuracy of the probe process requires the transmission of a large number of probe packets to provide measurements with good confidence. Furthermore, the schemes require a high multiplexing level on the links to make sure that the load variations are small compared to the average load.

A detailed comparison of the different endpoint admission control proposals is given in [44], showing that the performance of the different admission control algorithms is quite similar, and thus the complexity of the schemes may be the most important design consideration. The following sections briefly summarize the three main sets of proposals.

Admission Control Based on Probe Loss Statistics In the proposal from Karlsson *et al.* [36, 37, 45, 46] the call admission is decided based on the experienced packet loss during a short probe phase, ensuring that the loss ratio of accepted flows is bounded. Delay and delay jitter are limited by using small buffers in the network. Probe packets and data packets of accepted flows are transmitted with low and high priority respectively, to protect accepted flows from the load of the probe streams. The probing is done at the peak rate of the connection and the flow is accepted if the probe packet loss rate is below a predefined threshold. This procedure ensures that the packet loss of accepted flows is always below the threshold value.

Admission Control Based on ECN Marks The congestion level in the network in the proposal from F. Kelly *et al.* [39] and T. Kelly [40] is determined by the number of probe packets received with ECN marks by the end host. In this case, probe packets are transmitted together with data packets. To avoid network overload caused by the probes themselves the probing is done incrementally in probe rounds that last approximately one RTT, up to the peak rate of the incoming call. ECN–enabled routers on the transmission path set the ECN congestion experienced bit when the router detects congestion, e.g., when the buffer content exceeds a threshold or after a packet loss [38]. The call is accepted if the number of marked packets is below a predefined value. This proposal suggests that by running appropriate end–system response to the ECN marks, a low delay and loss network can be achieved.

The call admission control is coupled with a pricing scheme [47]. In this case users are allowed to send as much data as they wish, but they pay for the congestion they create (the packets that are marked). In this congestion pricing scheme the probing protocol estimates the price of a call, that is compared with the amount the end–system is willing to pay. The scheme does not provide connections with hard guarantees of service; it merely allows connections to infer whether it is acceptable to enter the network or not.

Admission Control Based on Delay Variations Bianchi *et al.* [43, 48] (first version published in [42]) propose to use measurements on the variation of packet inter–arrival time to decide about call admission, based on the fact that a non–negligible delay jitter can be observed even for accepted loads well under the link capacity. The admission control is designed to support IP telephony, thus it considers low and constant bit rate

flows. The probe packets are sent at a lower priority than data packets. The probing phase consists of the consecutive transmission of a number of probe packets with a fixed inter–departure time. A maximum tolerance on the delay jitter of the received probe packets is set at the receiving node, and the flow is rejected immediately if the condition fails for one probe packet. The maximum tolerance on the delay jitter and the number of probe packets transmitted regulates the maximum level of accepted load on the network links. This maximum load is selected in a way such that the packet loss probability and end–to–end delay requirements for the accepted calls are met.

4.4 PBAC: Probe-Based Admission Control

As an example, we discuss the endpoint admission control procedure based on packet loss statistics in detail [36, 37, 45, 46]. This solution offers a reliable upper bound on the packet loss for the accepted flows, while it limits the delay and delay jitter by the use of small buffers in the routers.

The admission control is done by measuring the loss ratio of probe packets sent at the peak rate of the flow and transmitted with low priority at the routers. The scheduling system of the routers consequently has to differentiate data packets from probe packets. To achieve this, two different approaches are possible. In the first one there are two queues, one with high priority for data and one with low priority for probe packets (see Figure 11). In the second approach, there is just one queue with a discard threshold for the probes. Considering the double–queue solution the size of the high priority buffer for the data packets is selected to ensure a low maximum queuing delay and an acceptable packet loss probability, i.e., to provide packet scale buffering [49]. The buffer for the probe packets on the other hand can accommodate one packet at a time, to ensure an over–estimation of the data packet loss. The threshold–queue can be designed to provide similar performance, as it is shown in [45], and the choice between the two approaches can be left as a decision for the router designer.

Figure 12 shows the phases of the PBAC session establishment scheme. When a host wishes to set up a new flow, it starts by sending a constant bit rate probe at the maximum rate the data flow will require. The probing time is chosen by the sender from a range of values defined in the service contract. This range forces new flows to probe for a sufficient time to obtain a sufficiently accurate measurement, while it prohibits them from performing unnecessarily long probes. The probe packet size should be small enough so that there are sufficient number of packets in the probing period to perform the acceptance decision. When the host sends the probe packets, it includes the peak bit rate and the length of the probe, as well as a packet and flow sequence number in the data field of each packet. With this information the end host can perform an early rejection, based on the expected number of packets that it should receive not to surpass the target loss probability. The probe contains a flow identifier to allow the end host to distinguish probes for different sessions. Since one sender could open more than one session simultaneously, the IP address in the probes is not enough to differentiate them. When the probe process finishes, the sender starts a timer with a value over two times the expected round trip time. This timer goes off in case the sender does not receive an acknowledgment to the probe. The timer allows the sender to infer that none of the probe packets went through or the acknowledgment packet with the acceptance

decision from the receiver got lost. The sender assumes the first scenario and backs off for a long period of time, still waiting for a possible late acknowledgment. In case a late acknowledgment arrives the sender acts accordingly and cancels the backoff process.

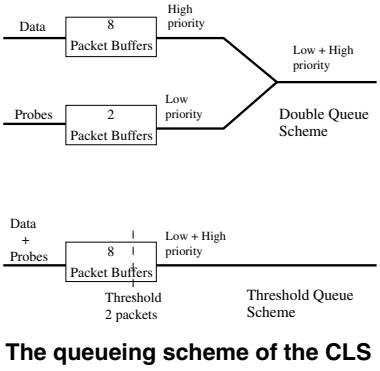


Fig. 11. The queuing system.

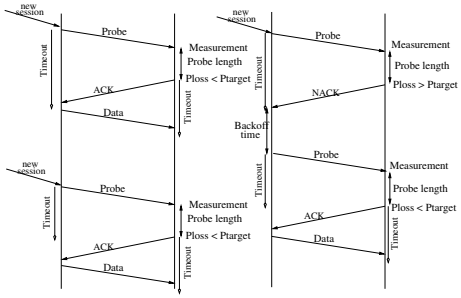


Fig. 12. The probing procedure.

Upon receiving the first probe packet for a flow, the end host starts counting the number of received packets and the number of lost packets (by checking the sequence number of the packets it receives). When the probing period finishes and the end host receives the last probe packet, it compares the probe loss measured with the target loss and sends back an acknowledgment packet accepting or rejecting the incoming flow. This acknowledgment packet is sent back at high priority to minimize the risk of loss. If the decision is positive, the receiver starts a timer to control the arrival of data packets. The value of this timer should be slightly more than two RTTs. If this timer goes off, the host assumes that the acceptance packet has been lost and resends it.

Finally, when the sending host receives the acceptance decision, it starts sending data with high priority, or, in the case of a rejection, it backs off for a certain amount of time, before starting to probe again. In subsequent tries, the sender host can increase the probe length, up to the maximum level allowed, so that a higher accuracy on the measurement is achieved. There is a maximum number of retries that a host is allowed to perform before having to give up. The back off strategy and the number of retries affect the connection setup time for new sessions and should be carefully tuned to balance the acceptance probability with the expected setup delay.

The acceptance threshold is fixed for the service class and is the same for all sessions. The reason for this is that the QoS experienced by a flow is a function of the load from the flows already accepted in the class. Considering that this load depends on the highest acceptance threshold among all sessions, by having different thresholds all flows would degrade to the QoS required by the one with the less stringent requirements. The class definition also has to state the maximum data rate allowed to limit the size of the sessions that can be set up. Each data flow should not represent more than a small fraction of the service class capacity (in the order of 1%), to ensure that statistical multiplexing works well.

A critical aspect of end–to–end measurement based admission control schemes is that the admission procedure relies on common trust between the hosts and the network. If this trust is not present, security mechanisms have to protect the scheme. First, as end hosts decide about the admission decision based on measurements performed in the network, their behavior has to be monitored to avoid resource misuse. Second, as information on the call acceptance has to be transmitted from the receiving node to the source, intruder attacks on the transmission path altering this information have to be avoided. The specific security mechanisms of the general end–to–end measurement based admission control schemes can be addressed in different ways that are out of the scope of this overview, but a simple cryptographic scheme has been proposed in [50].

Application to Multicast A solution to extend the idea of end-to-end measurement based admission control for multicast communication is presented in [46]. The proposed scheme builds on the unicast PBAC process. The admission control procedure assumes a sender–based multicast routing protocol with a root node (*rendez-vous* point) implemented. The root node of the multicast tree takes active part in the probing process of the admission control, while the rest of the routers only need to have the priority–based queuing system to differentiate probes and data, as in the original unicast PBAC scheme.

In order to adapt the admission control for multicast communication two multicast groups are created: one for the probe process and one for the data session itself. Senders first probe the path until the root node of the multicast tree, and start to send data if accepted by this node. The probe from the sender is continuously sent to the root node, and is forwarded along the multicast tree of the probe group whenever receivers have joined this group.

Receivers trying to join the multicast data group first join the probe group to perform the admission control. They receive the probe packets sent by the sender node and forwarded by the root node, and decide about the admission based on the packet loss ratio. If the call is accepted the receiver leaves the probe group and joins the data group. Consequently, receivers have to know the addresses of both the probe and the data multicast group to take part in the multicast communication.

The unicast PBAC scheme is thus extended for multicast operation without additional requirements on the routers. The procedure to join a multicast group is receiver initiated to allow dynamic group membership. The scheme is defined to support many simultaneous or non–simultaneous senders, and it is well suited to multicast sessions with a single multimedia stream or with several layered streams.

4.5 Summary

This chapter presents an overview on probe–based admission control schemes. These solutions provide call admission control for CLS–like services. The admission control process is based on actively probing the transmission path from the sender to the receiver and deciding about the call acceptance based on end–to–end packet loss, packet marking on delay jitter statistics. As only the end nodes, sender and receiver, take active part in the admission control process, these mechanisms are able to provide per–flow QoS guarantees in the current stateless Internet architecture.

5 A Component-Based Approach to QoS Monitoring

5.1 Introduction

The offering of Quality of Service (QoS) based communication services faces several challenges. Among these, the provisioning of an open and formalized framework for the collection and interchange of monitoring and performance data is one of the most important issues to be solved. Consider, for example, scenarios where multiple providers are teaming (intentionally or not) for the construction of a complex service to be sold to a final user, such as in the case of the creation of a Virtual Private Network infrastructure spanning multiple network operators and architectures. In this case, failure to provide certain required levels in the quality parameters should be met with an immediate attribution of responsibility across the different entities involved in the end-to-end provisioning of the service.

The same is also true in cases apparently much simpler, such as, for example, where a user requires a video streaming service across a single operator network infrastructure. In these situations there is also a need for mechanisms to measure the received quality of service across all of the elements involved in the service provisioning chain: the server system, the network infrastructure, the client terminal and the user application.

In described scenarios, the service and its delivery quality are negotiated through a contract, named the Service Level Agreement (SLA), between a user and a service provider. Such a service provider is intended as an entity capable to assemble service contents as well as to engineer network and server side resources. The subscription of a Service Level Agreement implies two aspects, only apparently unrelated: first, the auditing of the actual satisfaction of the current SLA with the service provider; second, the dynamic re-negotiation of the service level agreements themselves.

As far as the first aspect is concerned, we can reasonably forecast that as soon as communication services with QoS or other service-related guarantees (e.g. service availability) are available, and as soon as users start to pay for them, it will be required to verify whether or not the conditions specified in the SLA are actually met by the provider. With reference to the second aspect, indeed, re-negotiation of QoS has been always accepted as an important service in performance guaranteed communications, strongly connected to critical problems such as the efficiency of network resource allocation, the end-to-end application level performance, and the reduction of communication costs. For example we might consider a scenario where the quality of service received by a distributed application can be seen as influenced by several factors: the network performance, the server load and the client computational capability. Since those factors can be varying in time, it is logical to allow applications to modify Service Level Agreements on the basis of the QoS achievable and perceivable at the application layer. We therefore believe that the possibility to modify the existing QoS based agreements between the service provider and the final user will assume an important role in Premium IP networks. Such networks provide users with a portfolio of services thanks to their intrinsic capability to perform a service creation process while relying on a QoS-enabled infrastructure. In order to allow the SLA audit and re-negotiation a framework for the monitoring of the received Quality of Service is necessary.

In this document, we propose a novel approach to the collection and distribution of performance data. The idea which paves the ground to our proposal is mainly based on the definition of an information document, that we called *Service Level Indication* (SLI). The SLI-based monitoring framework is quite simple in its formulation; nevertheless it brings in a number of issues, related to its practical implementation, to its deployment in real-life scenarios, and to its scalability in complex and heterogeneous network infrastructures. Some of these issues will be highlighted in the following, where we will also sketch some possible guidelines for deployment, together with some pointers to potential innovative approaches to this complex task.

The document is organized as follows. The reference framework where this work has to be positioned is presented in Section 5.2 . In Section 5.3 we introduce QoS monitoring issues in SLA-based infrastructures. In Section 5.4 we illustrate the data export process. Section 5.5 explains some implementation issues related to the proposed framework. Finally, Section 5.6 provides some concluding remarks on the presented work.

5.2 Reference Framework

This section introduces the general architecture proposed for the dynamic creation, provisioning and monitoring of QoS based communication services on top of Premium IP networks [51][52]. Such an architecture includes key functional blocks at the user-provider interface, within the service provider domain and between the service provider and the network provider. The combined role of these blocks is to manage user's access to the service, to present the portfolio of available services, to appropriately configure and manage the QoS-aware network elements available in the underlying network infrastructure, and to produce monitoring documents on the basis of measurement data.

Main components of the proposed architecture are the following: (i) **Resource Mediator(s)**: it has to manage the available resources, by configuring the involved nodes. Each service can concern different domains and then different Resource Mediators. Now, the Resource Mediator also has to gather basic monitoring data and export it; (ii) **Service Mediator(s)**: it is in charge of creating the service as required from the user, using the resources made available by one or more Resource Mediators. It has to map the SLA from the Access Mediator into the associated Service Level Specification (SLS) [53] to be instantiated in cooperation with the Resource Mediator(s); (iii) **Access Mediator(s)**: it is the entity that allows the users to input their requests to the system. It adds value for the user, in terms of presenting a wider selection of services, ensuring the lowest cost, and offering a harmonized interface: the Access Mediator presents to the user the currently available services.

5.3 A Monitoring Document: The Service Level Indication

Computer networks are evolving to support services with diverse performance requirements. To provide QoS guarantees to these services and assure that the agreed QoS is sustained, it is not sufficient to just commit resources since QoS degradation is often unavoidable. Any fault or weakening of the performance of a network element may result in the degradation of the contracted QoS. Thus, QoS monitoring is required to track

the ongoing QoS, compare the monitored QoS against the expected performance, detect possible QoS degradation, and then tune network resources accordingly to sustain the delivered QoS. In SLA-based networks it becomes of primary importance the availability of mechanisms for the monitoring of service performance parameters related to a specified service instance. This capability is of interest both to the end-users, as the entities that 'use' the service, and to the service providers, as the entities that create, configure and deliver the service. QoS monitoring information should be provided by the network to the user application, by collecting and appropriately combining performance measures in a document which is linked to the SLA itself and which is conceived following the same philosophy that inspired the SLA design: i) clear differentiation of user-level, service-level and network-level issues; ii) definition of interfaces between neighboring roles/components; iii) definition of rules/protocols to appropriately combine and export information available at different levels of the architecture.

In Premium IP networks, the service provisioning is the result of an agreement between the user and the service provider, and it is regulated by a contract. The SLA is the document resulting from the negotiation process and establishes the kind of service and its delivery quality. The service definition stated in the SLA is understood from both the user and the service provider, and it represents the service expectation which the user can refer to. Such SLA is not useful to give a technical description of the service, functional to its deployment. Therefore, a new and more technical document is needed. The Service Level Specification document derives from the SLA and provides a set of technical parameters with the corresponding semantics, so that the service may be appropriately modeled and processed, possibly in an automated fashion. In order to evaluate the service conformance to specifications reported in SLA and SLS documents, we introduce a new kind of document, the Service Level Indication. By mirroring the hierarchical structure of the proposed architecture, it is possible to distinguish among three kinds of SLIs: (i) *Template SLI*, which provides a general template for the creation of documents containing monitoring data associated to a specific service; (ii) *Technical SLI*, which contains detailed information about the resource utilization and/or a technical report based on the SLS requirements. This document, which pertains to the same level of abstraction as the SLS, is built by the Resource Mediator; (iii) *User SLI*, i.e. the final document forwarded to the user and containing, in a friendly fashion, information about the service conformance to the negotiated SLA. The User SLI is created by the Service Mediator on the basis of the SLS, the Template SLI and the Technical SLI.

The service monitoring has to be finalized to the delivery of one or more SLI documents. In the SLI issue, multiple entities are involved, as network elements, content servers, and user terminals. Involving all these elements has a cost: due to the usage of both computational and network resources, needed for information analysis and distribution. This cost depends on both the number of elements involved and the information granularity. From this point of view, monitoring may be under all perspectives considered as a service, for which ad hoc defined pricing policies have to be specified and instantiated. More precisely, drawing inspiration from the concept of *metadata*, we might hazard a definition of monitoring as a *metaservice*, i.e. a 'service about a service'. This definition is mainly due to the fact that a monitoring service cannot exist on its own: monitoring is strictly linked to a preexisting service category, for which it

provides some value-added information. Therefore we will not consider a standalone monitoring service, but we will rather look at it as an optional clause of a traditional service, thus taking it into account in the SLA negotiation phase.

5.4 Data Export

In the context of SLA-based services the following innovative aspect has to be considered: in order to allow users, service providers and network operators to have information about QoS parameters and network performance the need arises to export data collected by measuring devices. To this purpose, the concept of data model has to be introduced. Such model describes how information is represented in monitoring reports. As stated in [54], the model used for exporting measurement data has to be flexible with respect to the flow attributes contained inside reports.

Since the service and its quality are perceived in a different fashion depending on involved actors (end user, service provider, network operator), there is a need to define a number of documents, each pertaining to a specific layer of the architecture, suitable to report information about currently offered service level. As far as data reports, we have defined a set of new objects aiming at indicating whether measured data, related to a specific service instance, is in accordance with the QoS level specified in the SLA.

With reference to our architecture, it is possible to identify the components responsible for the creation of each of the monitoring documents (Figure 13).

Such documents are then exchanged among the components as described in the following, where we choose to adopt a bottom-up approach:

1. at the request of the Service Mediator, the Resource Mediator builds the Technical SLI document on the basis of data collected by the measuring devices. The fields it contains are directly derived from those belonging to the SLS and are filled with the actual values reached by the running service. The resulting document is sent to the Service Mediator;
2. thanks to the Technical SLI received from the Resource Mediator, the Service Mediator is capable to evaluate the service quality conformance with respect to the requests formulated through the related SLS. It can be interested in such information both for its own business and in order to gather data for the creation of a complete report in case a user requests one;
3. at the user's request, the Service Mediator, exploiting data contained in a Technical SLI, produces a further report indicating the QoS level as it is perceived by the end user. The document it is going to prepare is derived from a service specific template (the so-called SLI Template), which provides an abstraction for the measurement results in the same way as the SLA Template does with respect to the service parameters. Such a document, hereby called User SLI, is ready for delivery to the end user;
4. the Access Mediator receives the User SLI from the Service Mediator, puts it in a format that is compliant with both the user's preferences and the user's terminal capabilities and forwards it to the end user.

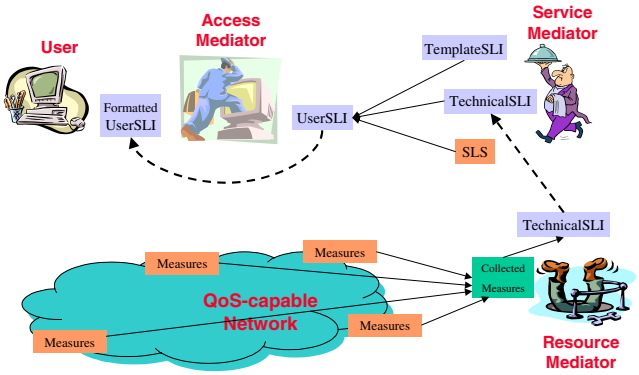


Fig. 13. Measurement data export.

5.5 Implementation Issues

Upper Layers A Service Mediator can add to the service offer the monitoring option. If this is the case, two different strategies are possible. In the first scenario, the Service Mediator evaluates the resource usage for monitoring service as a fixed cost. Such a cost will be taken into account when the quotation is prepared. This strategy does not modify the business process related to the specific service since the user's request to monitor its own SLA uniquely implies a fixed add-on to the service quotation. Such an add-on depends upon the business strategies of both the Service Mediator and the Resource Mediator.

In the second scenario, the Service Mediator shows two different quotations to the user: the first one related to the service instance selected by the user, the second one regarding the monitoring activity. This solution introduces a scenario in which the negotiation of the SLA monitoring metaservice is considered. Interactions between the Access Mediator and the Service Mediator can be formalized through the same business process as that describing "traditional" services, such as VoD, VPN, and VoIP. Such a business process includes the following business transactions:

1. checking the service availability
2. submitting a quotation request
3. accepting a service quotation
4. issuing the purchase order

The definition of business processes can benefit from the availability of a standard proposal coming from the electronic business research community, named ebXML [55]. A detailed description of the application of the ebXML framework to the mediation architecture may be found in [56].

Lower Layers In Section 5.4 we have defined the roles of components in the creation of models to export measurement data. In particular, it is the responsibility of the Service

Mediator to produce the User SLI intended to provide users with a high-level indication of the compliance (or not) of the delivered service to the negotiated SLA. The Resource Mediator, on the other hand, has to create the Technical SLI, which has the same fields as the corresponding SLS. These fields are filled by the Resource Mediator with the values resulting from the measurement activities.

At this point, the need arises to provide the Resource Mediator with monitoring data coming from the network devices so that it may be able to build the Technical SLI. In the proposed architecture the Resource Mediator is in charge of managing the whole underlying network for each domain (i.e. Autonomous System - AS) [57]. In particular, in the service configuration phase, it has to analyze the SLS, received from the Service Mediator, in order to select the subset of information related to its own domain. Such a subset is passed to the Network Controller, which translates it into a form that is compliant with the specific network architecture adopted (MPLS, Diffserv, etc.). The rest of the SLS is forwarded to the next peer Resource Mediator en-route towards the destination. If the user selects the monitoring option during the SLA negotiation, this choice does not modify the usual sequence of interactions concerning the service creation and configuration. In fact, once received from the Resource Mediator the subset of information contained in the SLS, the Network Controller translates it into a set of policy rules and, acting as a *Policy Decision Point* (PDP), sends policies to the underlying *Policy Enforcement Points* (PEPs), by exploiting the COPS protocol [58]. Upon the reception of a new policy, the PEP forwards it to the Device Controller, which produces configuration commands for the network device as well as rules enabling it to distinguish the traffic flow to be measured. Then, the Device Controller is able to appropriately configure the traffic control modules (e.g. allocation and configuration of queues, conditioners, markers, filters, etc.) and to perform the measurement activities.

Figure 14 depicts the policy framework components, distinguishing among the following abstraction layers: (i) **NI-DI**: Network Independent - Device Independent; (ii) **ND-DI**: Network Dependent - Device Independent; (iii) **ND-DD**: Network Dependent - Device Dependent.

Metering approaches Several solutions are available in order to allow Device Controllers to make measurements on the traffic flows at controlled devices.

Our solution is based on the use of the SNMP protocol, which permits to obtain performance data without any traffic injection (passive measurement). In this case, an SNMP client interacts with an SNMP agent located at the device in order to obtain measures about throughput and packet loss on a device interface. The main advantage of such a solution is the capability to obtain information about every network device that supports the SNMP standard protocol. If involved devices are routers or switches made by Cisco, it is possible to exploit a particular Cisco IOS functionality, named NetFlow.

Exporting data When the Device Controller, acting as a meter, obtains data about the controlled device, it has to forward it to the Resource Mediator. Using this raw data, the Resource Mediator can build the Technical SLI.

The Resource Mediator is able to create such an end-to-end document thanks to the interactions among the involved components, as described below:

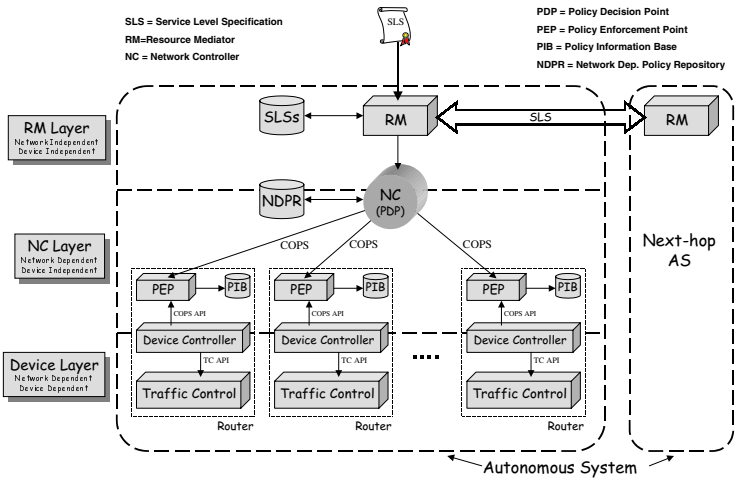


Fig. 14. Overview of the policy framework components.

1. The Device Controller sends the measurement results to the Policy Enforcement Point, e.g. using the COPS protocol [59]. This approach perfectly mirrors the policy-based architecture used at service configuration time.
2. Each PEP forwards received data to the Network Controller, which acts as a collector of measurement data from different measurement points;
3. Finally, the Resource Mediator, by gathering information from both its Network Controller and adjacent Resource Mediator, can build the Technical SLI. Such a document contains information about end-to-end performance metrics related to the traffic flow specified in the SLS.

5.6 Discussion and Conclusions

In this work, we have presented a component-based architecture for QoS measurement and monitoring. It is clear that the provisioning of such a feature is particularly complex and critical, since it involves the coordination and orchestrated operation of a large number of elements, separately owned and managed along what we have called the provisioning chain from the service location to the end user. We therefore foresee a number of issues to be faced: for some of them we believe a solution can be already provided, while for others the discussion is still open. We briefly mention here the main facets of the general issue of QoS monitoring, focusing on the networking infrastructure. First of all, the collection of monitoring data from the network elements. This issue is clearly related to both technical and business aspects. As far as the first ones, the work ongoing in the area of policy based management of network elements is providing a technical framework in which the control and configuration of network nodes will be much more straightforward than that currently achievable through the traditional SNMP based approach. However, it is clear that for global communication infrastructures involving a large number of nodes with a huge number of active connections we have a problem of

scalability with respect to the collection and delivery of performance data. In spite of this, we believe that there are features in the existing network architectures that might be exploited to reduce at least this problem. For example, in Diffserv based network architectures monitoring of Service Level Agreements can be performed usually per traffic classes and not per single traffic flows, and could be normally limited to the ingress and egress points of a domain. More detailed performance data collections (in terms of specific flows or network elements) could be triggered only in the presence of specific demands from the involved parties or in the case of anomalies. As far as the business aspects, i.e. those related to the business nature of the provisioning of communication services, we can mention here the one we believe is the most important: trust. In global networks, large scale infrastructures will be managed by a multitude of different operators, each managing a separate network domain. Quality of Service will therefore be an issue involving a number of parties, each responsible only for the service provided in the domain that it directly manages. Such parties will be obliged, at the same time, to compete and to cooperate with peering entities. Can we foresee a scenario where such performance data will be openly (albeit in a controlled way) available? We believe that rather than being an obstacle to the deployment of a common framework for SLA monitoring, trust will be an important trigger for it, if not a prerequisite. In fact, we can expect that no operator will start charging for premium services involving infrastructures owned by others without a formal, standardized way for exchanging performance data about the communication services offered to and received from other operators.

A further issue is related to the devising of a common quality of service measurement framework. It is clear that performance data should be provided in a way that is independent of both the network architecture offering the service and the application service demanding it. Our proposal is a first attempt in this direction.

6 Deterministic Delay Guarantees under the GPS Scheduling Discipline

6.1 Introduction

Under the GPS scheduling discipline traffic is treated as an infinitely divisible fluid. A GPS server that serves N sessions is characterized by N positive real numbers $\phi_1, ..., \phi_N$, referred to as weights. These weights affect the amount of service provided to the sessions (or, their bandwidth shares). More specifically, if $W_i(\tau, t)$ denotes the amount of session i traffic served in a time interval $(\tau, t]$ then the following relation will hold for any session i that is continuously backlogged in the interval $(\tau, t]$; session i is considered to be backlogged at time t if a positive amount of that session traffic is queued at time t.

$$\frac{W_i(\tau, t)}{W_j(\tau, t)} \geq \frac{\phi_i}{\phi_j}, j = 1, 2, ...N \tag{3}$$

The Generalized Processor Sharing (GPS) scheduling discipline has been widely considered to allocate bandwidth resources to multiplexed traffic streams. Its effectiveness and capabilities in guaranteeing a certain level of Quality of Service (QoS) to the

supported streams in both a stochastic ([60, 61, 62]) and deterministic ([63, 64, 65, 66]) sense have been investigated.

In this subchapter the single node case is considered and sessions (sources) are assumed to be (σ, ρ) leaky bucket constrained and to have a stringent delay requirement. Such sessions are referred to as QoS sensitive sessions in the sequel; the triplet (σ_i, ρ_i, D_i) is used in order to characterize the QoS sensitive session i, where σ_i, ρ_i and D_i represent the burstiness, the long term maximum mean arrival rate and the delay requirement of session i, respectively.

In the first part of the subchapter the problem of Call Admission Control (CAC) is considered. CAC plays a key role in provisioning network resources to meet the QoS requirements of traffic. It has the function of limiting the amount of traffic accessing the network and can have an important impact on network performance. A CAC algorithm is described which fully exploits the bandwidth sharing mechanism of GPS and determines the *optimal weights ϕ directly from the QoS requirements of the sessions*.

Nevertheless, the use of GPS scheduling discipline can lead to inefficient use of resources even if the optimal CAC scheme is employed. In the second part of the subchapter a service paradigm is described according to which each session is "represented" by two entities – referred to as session components – in the GPS server. Each session's component is assigned a weight and it is served by the GPS server. The work provided by the GPS server to the components of a session is mapped back to the original session. It turns out that the proposed service scheme leads to resource utilization improvement.

Related Work The GPS scheduling discipline has been introduced in [63], [64] where bounds on the induced delay have been derived for single node and multiple nodes systems, respectively. These (loose) delay bounds have a simple form allowing for the solution of the inverse problem, that is, the determination of the weight assignment for sessions demanding specific delay bounds, which is central to the Call Admission Control (CAC) problem.

Tighter delay bounds have been derived in [66] and in [65] (also reported in [67]). These efforts have exploited the dependencies among the sessions -due to the complex bandwidth sharing mechanism of the GPS discipline- to derive tighter performance bounds. Such bounds could lead to a more effective CAC and better resource utilization. The inverse problem in these cases, though, is more difficult to solve. For example, the CAC procedure presented in [65] employs an exhaustive search having performance bound calculations as an intermediate step. Specifically, the maximum delay experienced by the sessions is determined for a weight assignment and the assignment is modified trying to maximize an objective function. While the search in [65] terminates after a finite number of steps, it does not guarantee that an acceptable assignment does not exist if not found.

The fairness oriented nature of GPS scheduling discipline, which follows directly from its definition, makes GPS an appealing choice for a best effort environment, or more precisely for an environment where fairness is the main concern. Some arguments on why fairness is not necessarily the main concern even in a best effort environment may be found in [68] where some state dependent scheduling policies aiming to decrease loss and / or delay jitter by "sacrificing" fairness are presented.

Modifications / extensions of the GPS scheduling discipline include ([69], [70], [71]). In [69] Service Curve Proportional Sharing (SCPS) has been proposed. It is a generalization of GPS, according to which the (constant) weighting factors used by the GPS are replaced with well defined varying weights. In [70] a less complex implementation is proposed for the special case of piecewise linear service curves. In [71] a modification of the GPS scheduling discipline, aiming to improve the QoS provided to adaptive sessions is presented, according to which each session is assigned two weights; one weight determines its guaranteed rate and the other weight determines its target rate.

6.2 Optimal CAC Algorithm

The optimal CAC algorithm for the GPS scheduling discipline has been derived in [72] by considering a mixed traffic environment in which the bandwidth resource controlled by the GPS server is assumed to be shared by a number of QoS sensitive streams and best effort traffic. This system will be referred to as a Best Effort Traffic Aware Generalized Processor Sharing (BETA-GPS) system.[16] The Best Effort Traffic Aware (BETA) GPS system is depicted in figure 15. The BETA-GPS server capacity C_G is assumed to be shared by N QoS sensitive sessions with descriptors $(\sigma_i, \rho_i, D_i), i = 1, \ldots, N$ and best effort traffic represented by an additional session. Each session is provided a buffer and the input links are considered to have infinite capacity. Generally, the task

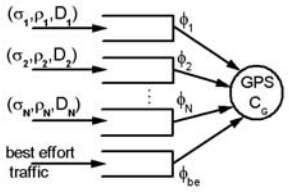


Fig. 15. The BETA-GPS system

of CAC is to determine whether the network can accept a new session without causing QoS requirement violations. In the case of a GPS scheduler it should also provide the server with the weight assignment which will be used in the actual service of the admitted calls. A CAC scheme for a GPS server is considered to be optimal if its incapability to admit a specific set of sessions implies that no ϕ assignment exists under which the server could serve this set of sessions (respecting all QoS requirements).

An optimal CAC scheme for the BETA-GPS system should seek to maximize the amount of service provided to the (traffic unlimited) best effort session under any arrival scenario and over any time horizon, while satisfying the QoS requirement of the (traffic limited) QoS sensitive sessions. That is, it should seek to maximize the normalized[17] weight assigned to the best effort traffic (ϕ_{be}), while satisfying the QoS requirement of

[16] In a system where only QoS sensitive sessions are present the existence of an extra session may be assumed and the presented algorithm be applied (see [72]).

[17] Without loss of generality, it is assumed that $\sum_{i=1}^{N} \phi_i + \phi_{be} = 1$

QoS sensitive sessions. Obviously, maximizing the weight assigned to the best effort traffic is equivalent to minimizing the sum of weights assigned to the QoS sensitive sessions.

Optimal CAC Scheme for the BETA-GPS System In the sequel only the rationale of the optimal CAC algorithm is described and a numerical example is provided; the detailed algorithm may be found in [72].

The CAC problem for a GPS system is simplified in view of the following Theorem (Theorem 3, [63]): If the input link speed of any session i exceeds the GPS service rate, then for every session i, the maximum delay D_i^* and the maximum backlog Q_i^* are achieved (not necessarily at the same time) when every session is greedy starting at time zero, the beginning of a system busy period. It is noted that a GPS system busy period is defined to be a maximal time interval during which at least one session is backlogged at any time instant in the interval. A session i is characterized as greedy starting at time τ if it sends the maximum allowable amount of traffic starting at time τ and an all-greedy GPS system is defined as a system in which all the sessions are greedy starting at time 0, the beginning of a system busy period.

The aforementioned theorem implies that if the server can guarantee an upper bound on a session' s delay under the all greedy system assumption this bound would be valid under any (leaky bucket constrained) arrival pattern. Thus, in the framework of the CAC problem it is sufficient to examine only all greedy systems.

It [72] it has been shown that a given acceptable ϕ assignment is converted to the optimal one if each QoS sensitive session's busy period is expanded as much as its QoS would permit, starting from the set of QoS sensitive sessions that empty their backlog first in order.[18] This implies that in order to determine the optimal ϕ assignment it is sufficient to allocate to the QoS sensitive sessions such weights that their QoS would be violated if their busy periods were expanded.

In order to determine these optimal weights the CAC algorithm of [72] emulates the all greedy system and examines the QoS sensitive sessions at all time instants coinciding with either the delay bound or the backlog clearing time of some session; these time instants are referred to as checkpoints in [72]. For each QoS session the algorithm computes two quantities (two potential weights); the minimum weight that is necessary for the QoS requirements of the session to be met *up* to the specific time instant and the weight that is necessary for the QoS requirements of the session to be met *after* the specific time instant, denoted as $\phi_i^-(\tau_j)$ and $\phi_i^+(\tau_j)$ at the checkpoint τ_j ,respectively. If $\phi_i^-(\tau_j) \geq \phi_i^+(\tau_j)$ session i is assigned a weight $\phi_i = \phi_i^-(\tau_j)$, since this is the minimum weight that could be assigned; else the specific session is examined again at the next checkpoint. In order to apply the aforementioned procedure the algorithm keeps track of the bandwidth that is available to the still backlogged sessions (a greedy (σ_i, ρ_i) constrained session requires a rate equal to ρ_i after it empties its backlog, and

[18] A ϕ assignment is characterized as acceptable if it is feasible (that is $\sum_{i=1}^{N} \phi_i < 1$)and delivers the required QoS to each of the supported QoS sensitive sessions; a ϕ assignment is characterized as more efficient than another if the sum of ϕ's $\sum_{i=1}^{N} \phi_i$ under the former assignment is smaller than that under the latter.

thus, the additional (compared to ρ_i) bandwidth that the session utilizes until it empties its backlog is shared among the still backlogged sessions in proportion to their weight).

Next a numerical example is provided, where the optimal CAC scheme for the BETA-GPS system is compared with the effective bandwidth-based CAC scheme. Deterministic effective bandwidth ([73]) can be used in a straightforward way to give a simple and elegant CAC scheme for the GPS scheduler. A similar approach is followed in [62] for the deterministic part of their analysis. The deterministic effective bandwidth of a (σ_i, ρ_i, D_i) session is given by $w_i^{eff} = \max\{\rho_i, \frac{\sigma_i}{D_i}\}$. It is easy to see that the requirements of the QoS sensitive sessions are satisfied if they are assigned weights such that $\phi_i C_G = w_i^{eff}$ ($\frac{\phi_i}{\phi_j} = \frac{w_i^{eff}}{w_j^{eff}}, \forall i, j \in QoS$).

Two traffic mixes are considered which are denoted as *Case 1*and *Case 2* in Table 1 where the parameters of the sessions for each case are provided. All quantities are considered normalized with respect to the link capacity C. In order to compare the optimal

Table 1. Sessions under investigation

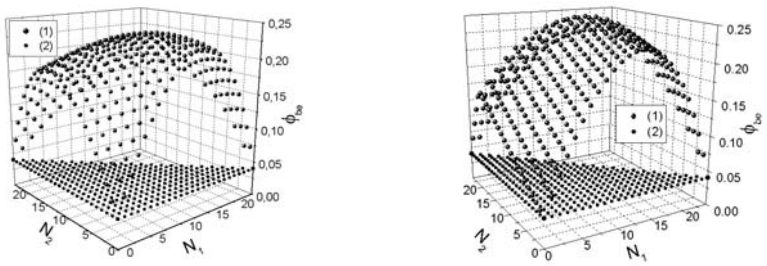
Case 1	s_1	s_2		s_3
Case 2	s_1		s_2	s_3
σ_i	0.04	0.16	0.04	0.64
ρ_i	0.01	0.01	0.04	0.01
D_i	1	4	4	16
w_i^{eff}	0.04	0.04	0.04	0.04

CAC algorithm with the effective bandwidth-based CAC scheme the following scenario is considered. The effective bandwidth-based CAC scheme admits the maximum number of sessions under the constraint that a nonzero weight remains to be assigned to best effort traffic. From Table 1 it can be seen that the effective bandwidth of each QoS sensitive session is 1/25 of the server's capacity (which is considered to be equal to the link capacity ($C_G = C$)), implying that for the BETA-GPS system at most 24 QoS sensitive sessions can be admitted under the effective bandwidth-based CAC scheme. This means that $N_1 + N_2 + N_3 = 24$ must hold and that the best effort traffic is assigned weight equal to 0.04 for each such triplet (N_1, N_2 and N_3 denote the number of admitted sessions of type s_1,s_2 and s_3 respectively).

For each triplet (N_1, N_2, N_3), $N_1 + N_2 + N_3 = 24$, the weight assigned to the best effort traffic by the optimal CAC scheme is computed. The results are illustrated in figure 16.

6.3 Decomposition-Based Service (DS-) Scheme

According to the Decomposition-based Service (DS-) scheme, which is depicted in figure 17, sessions are not served directly by the GPS scheduler. The GPS scheduler is employed by a Virtual System which is provided with an exact replica of each session's traffic. In the Virtual System some of the sessions (replicas) are decomposed into two components, by engaging properly dimensioned leaky buckets, one for each session.



Case 1 *Case 2*

Fig. 16. Weight assigned to the best effort traffic according to the (1) optimal CAC (2) effective bandwidth-based CAC scheme, both under the constraint $N_1 + N_2 + N_3 = 24$. The minimum guaranteed rate to the best effort traffic is $\phi_{be} C_G$

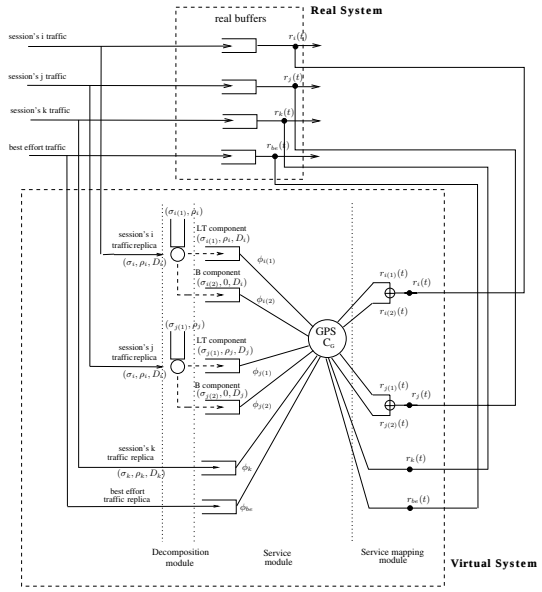


Fig. 17. The DS-system

These two components are buffered in separate virtual buffers and are assigned weights in proportion to which they are served by the GPS scheduler. The real traffic of a session is served at any time instant at a rate equal to the sum of the service rates of its components in the Virtual System.

The Virtual System is necessary (that is, the decomposition can not be applied directly on the original sessions) so that the real traffic of each session remains in a common buffer in order to avoid "packet reordering"; that is, to ensure that each session's

traffic leaves the system in the order it arrived. The Virtual System may be considered to consist of three modules (see figure 17): (a) the decomposition module responsible for the decomposition of the replicas of the sessions (b) the service module responsible for determining the service provided to each session's components, and (c) the service mapping module which maps the service provided to the sessions components back to the original sessions.

Each session $i \sim (\sigma_i, \rho_i, D_i)$ may be considered as the superposition (aggregation) of two component sessions $i_{(1)} \sim (\sigma_{i(1)}, \rho_i, D_i)$ and $i_{(2)} \sim (\sigma_{i(2)}, 0, D_i)$, $\sigma_{i(2)} = \sigma_i - \sigma_{i(1)}$, which will be referred to as the Long Term (LT) and the Bursty (B) component of session i, respectively.

For the decomposition of session i a $(\sigma_{i(1)}, \rho_i)$ leaky bucket, with $\sigma_i > \sigma_{i(1)} > 0$, is employed. Session i traffic (replica) traverses the $(\sigma_{i(1)}, \rho_i)$ leaky bucket; the part of the traffic that finds tokens is considered to belong to the LT-component of session i and the rest of session's traffic is considered to belong to the B-component of the session. Both components of session i have t he same delay constraint D_i as the original session i.

It is noted that not all the sessions are necessarily decomposed into two components; only the sessions that fulfill the criteria described in Section 6.3 are decomposed (and $\sigma_{i(2)}$, $\sigma_{i(1)} = \sigma_i - \sigma_{i(2)}$, are determined). Sessions which are not decomposed are represented by only one component (session traffic replica) in the service module of the Virtual System.

CAC for the DS-System The following claim, whose proof may be found in [74] shows that in order to develop a CAC scheme for the DS-system it suffices to develop a CAC scheme for the Virtual System, since a schedulable traffic mix for the Virtual System is also schedulable for the DS-system whose central element is the Virtual System itself.

Claim. If the components of a session are served by the service module of the Virtual System in such a way that their QoS requirements are satisfied and the real session is served at any time instant at a rate equal to the sum of the service rates of its components, then the QoS requirements of the session are also satisfied.

Let $\mathcal{T}$ denote the original traffic mix requesting service by the DS-system. Let $\mathcal{T}'$ denote the traffic mix served by the service module of the Virtual System; $\mathcal{T}'$ is obtained from $\mathcal{T}$ by replacing sessions whose replicas are decomposed in the decomposition module of the Virtual System by their components; the service module of the Virtual System serving $\mathcal{T}'$ is equivalent to (does not differ in any way from) the BETA-GPS system serving $\mathcal{T}'$ (see Figures 15 and 17). Thus, the Virtual System may be considered as a BETA-GPS system with a tunable input (under the constraints that the envelopes of the sessions components have a predefined form and the sum of the envelopes of each session components is equal to the envelope of the original session). Consequently, the CAC scheme for the Virtual System (or equivalently the DS-system) may be considered as a CAC scheme for a system similar to the BETA-GPS system with an extra degree of freedom (an extra dimension in the design space), which is the ability to determine sessions components. The CAC scheme for the DS-system that has been derived in

Table 2. Sessions parameters

session id i	1	2	3
σ_i	10.95	5.45	10.95
ρ_i	0.05-1	0.05-1	0.05-1
D_i	12	24	36

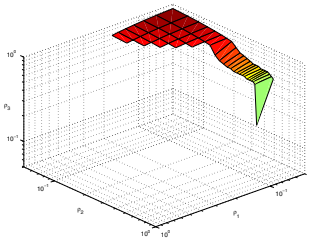


Fig. 18. BETA-GPS system employing the optimal CAC scheme for the BETA-GPS

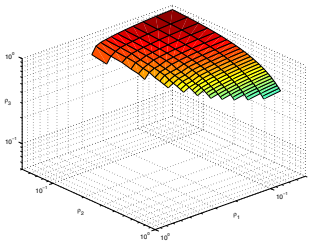


Fig. 19. DS-system employing the D_CAC scheme

[74] and is referred to as the D_CAC scheme (Decomposition-based Call Admission Control scheme) is a generalization of the optimal CAC scheme for the BETA-GPS, in the same way as the DS-system is a generalization of the BETA-GPS system. It may be considered as consisting of two distinct functions: (a) one responsible for determining which sessions of the original traffic mix will be decomposed in the Virtual System and the exact form of their components, and (b) one responsible for determining the weight assignment for the sessions components. The optimal CAC scheme for the BETA-GPS is employed for the determination of the weight assignment of the session components. The other function, which is responsible for determining the session components is such that a session replica is decomposed in the Virtual System only if (and in such a way that) this decomposition: (a) leads to better (or at least not worse) resource utilization, and (b) the session is assigned a total weight (sum of the weights of the session components) no greater than if it were not decomposed.

These conditions imply that the decomposition procedure is such that it leads to a traffic mix $\mathcal{T}'$ – starting from a traffic mix $\mathcal{T}$ – for which the optimal CAC scheme for the BETA-GPS returns total weights for the components of any session that are typically smaller and never larger than those identified by applying the optimal CAC scheme for the BETA-GPS to the original traffic mix $\mathcal{T}$. This allows a typically greater and never smaller weight to be assigned to the best effort traffic, for the traffic mixes that are schedulable in the BETA-GPS system. In addition, some traffic mixes that are not schedulable by the GPS in the BETA-GPS system, can be admitted in the DS-system (whose service module is a GPS server), and thus, the transformation of $\mathcal{T}$ to $\mathcal{T}'$ may be considered to lead to an *indirect* expansion of the schedulability region of GPS.

The details of the D_CAC scheme may be found in [74]. In the sequel a numerical example is provided that demonstrates the improvement in resource utilization that can be achieved by employing the DS-system. The considered traffic mix consists of three

sessions whose parameters are shown in Table 2. σ_i and D_i are kept constant while ρ_i takes values between 0.05 and 1 (the step is equal to 0.02). The link capacity is assumed to be equal to 1. In figure 18 the maximum long term rate of session 3 (ρ_3), such that the traffic mix is schedulable, is depicted as a function of ρ_1 and ρ_2 for the BETA-GPS system employing the optimal CAC scheme for the BETA-GPS . In figure 19 the corresponding plot is given for the case of the DS-system employing the D_CAC scheme.

7 MPEG Traffic Modeling at the Frame and GoP Levels Using GBAR and D-BMAP Processes

7.1 Introduction

Modern broadband telecommunications networks are intended to deliver traffic generated by multimedia applications. Video-on-Demand (VoD) is an example of such applications. Traffic generated by VoD servers must be delivered to destinations using a certain network technology. To provide an adequate quality of service (QoS), some quality control mechanisms have to be implemented in the network.

It is well known that uncompressed video information can easily throttle the available bandwidth. In order to achieve efficient transmission of such traffic, video information must be compressed and encoded in accordance with one of the existing compression algorithms. Today, one of the most used algorithms is MPEG.

Modeling of variable bit rate (VBR) video traffic has become an important issue since it provides the starting point for both theoretical analysis and engineering design. The source models of different types of video traffic are needed to design and study the network performance and also to predict the QoS that a particular video application may experience at different levels of network congestion [75, 76].

A number of video source models have been proposed in literature. Recently, most attention has been paid to MPEG traffic modeling using discrete- and continuous-time Markov chains [77, 76]. Such models produce an excellent approximation of both the histogram of frame sizes and the autocorrelation function (ACF) of the empirical data. However they are computationally inefficient because the construction of these Markov modulated processes from empirical data involves the so-called inverse eigenvalue problem [77, 76]. This drawback restricts their use in simulation studies where it is necessary to produce the model in real time.

First, we propose a novel frame level two step MPEG modeling algorithm that emulates the behavior of a single MPEG-1 elementary video stream. The proposed algorithm captures well the distribution of frame sizes and the ACF at the frame level. Then, based on the special case of the D-BMAP process, we model the smoothed traffic from a single MPEG source. Based on the statistical analysis of MPEG traffic at the group of pictures (GoP) level we propose to limit the state space of the modulating Markov chain of the D-BMAP (discrete-time batch Markovian arrival process) such that it has only four states. The limited state space allows us to decrease the complexity of the algorithm with the necessary accuracy being retained. Our D-BMAP model is simple, captures both the histogram of relative frequencies and the ACF of a single smoothed

MPEG traffic source, allows an analytical evaluation of the queuing systems and can be used in simulation studies.

It should be noted that in modeling MPEG traffic we focus on MPEG data, ignoring auxiliary information: the sequence header, packet headers etc. The rest of the section is organized as follows. In the next subsection we briefly outline MPEG traffic. Then we consider our modeling methodology. In Section 7.3 we present a GBAR(1) (gamma-beta autoregressive process of order 1) MPEG traffic model. Section 7.4 provides an MPEG traffic methodology based on the D-BMAP process. Conclusions are drawn in the last section.

7.2 MPEG Traffic

MPEG traffic is characterized by high peak rates and drastic short-term rate changes. These features cause losses within node buffers or remarking within traffic conditioners. To deal with these problems, we assume that the MPEG traffic is smoothed at the group of pictures (GoP) level as follows:

$$K^m(h) = \frac{1}{m} \sum_{i=(h-1)m+1}^{hm} X(i), m = 1, l_{GoP}, ..., N, h = \frac{n}{m}, \frac{n}{m} - 1, ..., 1, \quad (4)$$

where *N* is the number of frames, l_{GoP} is the length of the GoP, *X(n)* the sizes of individual frames and $K^m(h)$ the sizes of the smoothed sequence. In our study the GoP has a 12,3,2 structure [78] and we can only use those smoothing patterns whose length is divisible by 12. We therefore set *m* equal to 12.

The outlined approach does not require any computational resources during the transmission of video, provides a deterministic delay and efficiently uses the frame structure of MPEG traffic. This approach is therefore suitable for use in VoD systems.

7.3 VoD Traffic Models

The GBAR(1) modeling algorithm has a two step structure. Heyman in [79] shows that the model of MPEG traffic at frame level must incorporate three strongly auto- and cross-correlated processes. These are the *I*-frame process, the *P*-frame process and the *B*-frame process. Therefore, we should take into account the correlation structure of empirical data. At the first step of the proposed algorithm we use the property that *I*-frames are coded autonomously without reference to preceding or following frames, and, therefore we can state that the sizes of *I*-frames are independent of the sizes of *P*- and *B*-frames. Based on this we propose to model the *I*-frame generation process as an independent stochastic process. In order to accomplish this we use the GBAR(1) process. The sequence of sizes of *I*-frames that is obtained during the first step is used together with intra-GoP correlation information as initial data for the second step of the algorithm.

Before discussing the details of the algorithm we point out the necessary prerequisites. As has been shown in [80] the distribution of intra-frame sizes at the output of an H.261 codec can be approximated by a negative-binomial distribution and its continuous equivalent, the gamma distribution. The intra-frame coding scheme utilized by

H.261 codecs is almost identical to that used for *I*-frame coding in the MPEG standard. Therefore, we can expect that the gamma distribution will provide a good approximation for the *I*-frame sizes. To prove this we compared an empirical *I*-frame size distribution and a gamma distribution for the pattern *"Star Wars"* by quantile-quantile statistical analysis. We used the following parameter values: the shape parameter was set to 3.0 and scale parameter was set to 2.6*E*-5. These parameters were derived directly from the empirical distribution. We have noticed in [81] that such an approach gives a good approximation for the empirical data.

The property that allows us to proceed from the *I*-frame generation process to an MPEG aggregate frame generation process is a strong dependence between the *I*-frame sizes and *B*- and *P*-frame sizes within the same GoP. This property was discovered by Lombardo *et al.* [82, 83], which is also explained in [77] and later was called "intra-GoP correlation".

***I*-frame Process** In order to represent the *I*-frame generation process we propose the GBAR(1) process. It was originally presented by Heyman [79], where it was used as the approximation model for the frame size distribution of the H.261 codec. The main distinctive feature of the process appears in the geometrical distribution of its ACF. This property allows us to model the ACF of the empirical data that exhibits short range dependence (SRD) behavior. Moreover, the marginal distribution of the frame size sequence is a gamma distribution.

Let $G(\beta, \lambda)$ be a random variable with a gamma distribution with shape parameter β and scale parameter λ, and let $B(p, q)$ be a random variable with a beta distribution with parameters p and q. The GBAR(1) process is based on two well-known results: the sum of independent random variables $G(\alpha, \lambda)$ and $G(\beta, \lambda)$ is a $G(\alpha + \beta, \lambda)$ random variable, and the product of independent random variables $B(\alpha, \beta - \alpha)$ and $G(\beta, \lambda)$ is a $G(\alpha, \lambda)$ random variable.

Thus, if we denote $G(\beta, \lambda) = X_{n-1}$, $A_n = B(\alpha, \beta - \alpha)$ and $B_n = G(\beta - \alpha), \alpha$ and if they are mutually independent, then $X_n = A_n X_{n-1} + B_n$ is a stationary stochastic process $\{X_n, n = 0, 1, ...\}$ with marginal distribution $G(\beta, \alpha)$. The ACF is geometrical and given by $r_k = (\alpha/\beta)^k$. Parameters β and λ can be directly estimated from empirical data. Assume that the ACF is above zero for sufficiently large lag *k*: $r_k = p^k$ then the following equation holds $\alpha = p\beta$ and α is obtained.

Random variables with gamma or beta distributions generate non-integer values because they are continuous. Since the number of bits in the frame is a discrete random variable we round the values obtained from this process to the nearest integer.

Approximation of Intra-GoP Correlation In order to approximate the intra-GoP correlation structure and to obtain the sizes of *P*- and *B*-frames, the algorithm proposed in [77] can be used. The algorithm is intended to clarify the dependency between the mean value and the standard deviation of *I*-frames and the sizes of appropriate *P*- or *B*-frames. These dependencies are determined as follows:

$$M[X] = f^M(K_I), \sigma[X] = f^\sigma(K_I), X \in \{P, B\}, \tag{5}$$

where $M[X]$ is the mean value of the appropriate frame (*P* or *B*), $\sigma[X]$ is the standard deviation of the *P*- or *B*-frames, and K_i is the size of the I-frame. Results are shown in [77, 81, 82, 83] where it was shown that these dependencies can be approximated by straight lines.

The mean value and the standard deviation given by (5) serve as initial parameters for certain probability distributions. These probability distributions will allow us to obtain *B*- and *P*-frame sizes holding the *I*-frame size constant (or, more precisely, an interval of *I*-frame sizes). Note that in this case, the output values will vary even for constant *I*-frame size. This property of the model emulates the behavior of the real codecs.

Approximation of *B*- and *P*-frame Sizes It has been shown that histograms of the *B*- and *P*-frame sizes corresponding to a certain *I*-frame size can be approximated by a gamma distribution [82, 83]. The ACF of the *P*- and *B*-frame generation processes holding the *I*-frame size constant have the SRD property only. Since the GBAR process captures well all of the properties mentioned here we use this process as a model for *P*- and *B*-frame generation processes [81].

7.4 Traffic Modeling at the GoP Level

To model the smoothed traffic at the GoP level from a single MPEG source we propose to use the D-BMAP process. In each state of the modulating Markov chain of this process we can use an arbitrary distribution and the ACF of the process can be made to constitute a good approximation of the empirical ACF.

Consider general characteristics of the D-BMAP process. Let $\{W(b), n = 0, 1, ...\}$ be the D-BMAP arrival process. In accordance with D-BMAP, the number of arriving packets in each interval is modulated by an irreducible aperiodic discrete-time Markov chain (DTMC) with *M* states. Let *D* be the transition matrix of this process. Assume that the stationary distribution of the modulating DTMC is given by the vector $\overrightarrow{\pi}$. We define the D-BMAP process as a sequence of matrices $D(= k), k = 0, 1, ...$, each of which contains probabilities of transition from state to state with $k = 0, 1, 2, ...$ arrivals respectively [84]. Note that the number of arrivals in real models is always bounded.

Let the vector $\overrightarrow{G} = (G_1, G_2, ..., G_M)$ be the mean rate vector of the D-BMAP process. The input rate process of the D-BMAP $\{W(n), n = 0, 1, ...\}$ is defined by $\{G(n), n = 0, 1, ...\}$ with $G(n) = G_i$ while the Markov chain is in the state *i* at the time slot *n* [81]. The ACF of the rate process is given by [84]:

$$R^G(i) = \sum_{l, l \neq 1} \varphi_l \lambda_l^i,$$

$$\varphi_l = \overrightarrow{\pi} \left(\sum_{k=1}^{\infty} kD(= k) \right) \overrightarrow{g_l} \cdot \overrightarrow{h_l} \left(\sum_{k=1}^{\infty} kD(= k) \right) \overrightarrow{e}, i = 1, 2, ..., \qquad (6)$$

where λ_i is the i^{th} eigenvalue of D, $\overrightarrow{g_l}$ and $\overrightarrow{h_l}$ are the eigenvectors of *D* and $\overrightarrow{e}$ is a vector of ones.

Note that the ACF of the rate process consists of several geometric terms. Such behavior produces a good approximation of empirical ACFs that exhibit near geometrical sum decays [77, 85]. The number of geometrical terms composing the ACF depends on the number of eigenvalues which, in turn, depends on the number of states of the DTMC [86]. Thus, by varying the number of states of the modulating Markov chain we can vary the number of geometrical terms composing the ACF.

ACF Approximation In order to approximate the empirical ACF of MPEG traffic at the GoP level from a single MPEG source we use the method originally proposed in [77]. Particularly, we minimize the error of the ACF approximation γ by varying the values of the coefficients $(\lambda_i, \varphi_i), l = 1, 2, ..., K$ for each $K = 1, 2, ...$ in accordance with:

$$\gamma(k) = \frac{1}{i_0} \sum_{i=1}^{i_0} R^{emp}(i) - \frac{\sum_{i=1}^{K} \varphi_i \lambda_i}{R^{emp}(i)}, K = 1, 2, ..., \tag{7}$$

where *i* is the lag, i_0 is the lag when the empirical ACF reaches the confidence interval, and $R^{emp}(i)$ is the value of the ACF for lag *i*.

We note that we do not consider here those cases when $K > 3$. This is because through increasing the number of coefficients, which approximate the empirical ACF, the number of eigenvalues also increases and, therefore, the state space of the modulating Markov chain expands significantly. Thus, it is wise to keep the state space as small as possible and, from this point of view, $K = 2$ presents the best trade-off between the accuracy of the approximation of the empirical ACF and the simplicity of the modulating Markov chain. Note that with $K = 2$ the number of states of modulating Markov chain of the D-BMAP process may not exceed three.

At this stage the only approximation error is induced. This is the error of ACF approximation $\gamma(2)$ by two geometrically distributed terms.

Approximation by the Input Rate Process The construction of Markov modulated processes from empirical data involves a so-called inverse eigenvalue problem. It is known that a general solution of this problem does not exist. However, it is possible to solve such problems when some limitations on the form of the eigenvalues are set [77, 76].

Our limitation is that the eigenvalues should be located in the $(0, 1]$ fraction of the *X* axis. Note that one part of limitation $-1 \leq \lambda_l \leq 1, \forall l$ is already fulfilled since all eigenvalues of the one-step transition matrix of an irreducible aperiodic Markov chain are located in the $[-1, 1]$ fraction of *X* axis [86]. The second part $0 < \lambda_l \leq 1, \forall l$ should be fulfilled by the solution of the inverse eigenvalue problem.

We propose to construct the D-BMAP arrival process from two simple D-BMAP processes with a two-state modulating Markov chain. Let $\{W(n), n = 0, 1, ...\}$ be the D-BMAP arrival process, which models the smoothed traffic from an MPEG source, and $\{W_{emp}(n), n = 0, 1, ...\}$ be the empirical process of GoP sizes. $\{W^{[1]}(n), n = 0, 1, ...\}$ and $\{W^{[2]}(n), n = 0, 1, ...\}$ are two simple two-state D-BMAP processes (switched D-BMAP, SD-BMAP).

It is known that the rate process of a simple SD-BMAP can be statistically characterized by the triplet $(E[W], \varphi, \lambda)$ [87], where $E[W]$ is the mean arrival rate of the process, φ is the variance of the D-BMAP process and λ is the real eigenvalue of the modulating Markov chain which is not zero [86]. However, in order to define the rate process of the SD-BMAP, we should provide four parameters $(G_1, G_2, \alpha, \beta)$, where G_1 and G_2 are the mean arrival rates in state 1 and state 2 respectively, α is the probability of transition from state 1 to state 2 and β is the probability of transition from state 2 to state 1.

If we choose G_1 as the free variable [77] with constraint $G_1 < E[W^{[1]}]$ to satisfy $0 < \lambda \leq 1$ [87], we can obtain the other variables from the following equations [77, 87]:

$$G_2 = \frac{\varphi}{E[W] - G_1} + G_1, \alpha = \frac{(1-\lambda)(E[W] - G_1)}{G_2 - G_1}, \beta = \frac{(1-\lambda)(G_2 - E[W])}{G_2 - G_1} \quad (8)$$

Therefore, if we set $\lambda^{[1]} = \lambda_1$, $\lambda^{[2]} = \lambda_2$, $E[W] = E[W^{[1]}] + E[W^{[2]}]$, and choose $G_1^{[1]}$ and $G_1^{[2]}$ such that $G_1^{[1]} < E[W^{[1]}]$ and $G_1^{[2]} < E[W^{[2]}]$ we get both SD-BMAP arrival processes whose superposition gives us the D-BMAP process with the same ACF and mean arrival rate as the empirical data. The one-step transition matrix of the superposed process is given by the Kronecker product of composing processes $D = D^{[1]} \otimes D^{[2]}$.

It is clear from (8) that there is a degree of freedom when we choose the parameters $G_1^{[1]}$ and $G_1^{[2]}$. Moreover, the additional degree of freedom arises when we choose the values of $E[W^{[1]}]$ and $E[W^{[2]}]$. Thus, there is an infinite number of D-BMAP arrival processes with the same mean arrival rate $E[W^{emp}]$ which approximate the empirical ACF with error γ.

We also note that from the definition of the Kronecker product it follows that the eigenvalues of the matrix *D* which is the Kronecker product of matrices $D^{[1]}$ and $D^{[2]}$ with respective eigenvalues $\lambda_l^{[1]}, l = 1, 2, .., m$, and $\lambda_l^{[2]}, l = 1, 2, .., n$, has eigenvalues which are given by $\lambda_i = \lambda_l^{[1]} \lambda_l^{[2]}, i = 1, 2, .., n, .., nm$. Therefore, there is an additional error in the empirical ACF approximation, which is caused by only one eigenvalue $\lambda_l^{[1]} \lambda_l^{[2]}$ of the superposed process, since the one-step transition matrix of a two-state irreducible aperiodic Markov chain possesses two eigenvalues and one of them is always 1. Therefore, the error of the ACF approximation can be expressed precisely:

$$\gamma(2) = \frac{1}{i_0} \sum_{i=1}^{i_0} R^{emp}(i) - \frac{\sum_{l=1}^{3} \varphi_l \lambda_l^i}{R^{emp}(i)} \quad (9)$$

where $\lambda_3 = \lambda_1^{[1]} \lambda_1^{[2]}$.

Approximation of the Relative Frequencies of the GoP Sizes Note that the above mentioned derivation of the D-BMAP process restricts us to the mean arrival rate and the ACF and does not take into account the histogram of relative frequencies of GoP sizes. To assure that both the histogram and ACF are matched we should assign the PDF of GoP sizes to each state of the 4-state modulating DTMC such that the whole PDF matches the histogram of the GoP sizes.

Assume that the histogram of relative frequencies has *m* bins. Therefore, each PDF in each state of the modulating Markov chain should have not less than *m* bins. Since the stationary probabilities of the modulating Markov chain of the D-BMAP process are known for each PDF the following set of equations should hold:

$$\sum_{i=1}^{4} \left(f_j^i(\Delta)\pi_j^i \right) = f_j^{emp}, \sum_{j=1}^{m} \left(f_j^i(\Delta) j\Delta \right) = G^i, \sum_{j=1}^{m} f_j^i(\Delta) = 1, \qquad (10)$$

where $j = 1, 2, .., m$, $i = 1, 2, 3, 4, m$ is the number of histogram bins, $f_j^{emp}(\Delta)$ and $f_j^i(\Delta)$ are the relative frequency and probability respectively corresponding to the j^{th} bin in the i^{th} state of the modulating Markov chain, G^i is the mean arrival rate in state *i* and Δ is the length of the histogram intervals.

Note that we have only $(4 \cdot 2 + m)$ equations while there are $4m$ unknowns. We also should note that in general, if the Markov chain has *M* states and there are m histogram bins the number of unknowns is M_m and we have only $2M + m$ equations. It is seen that by increasing the number of states of the modulating Markov chain the complexity of the task increases rapidly. This is the additional reason why we should keep the state space of the modulating Markov chain as small as possible.

In order to get values of $f_j^i(\Delta), i = 1, 2, 3, 4, j = 1, 2, .., m$, we propose to use a random search algorithm. In accordance with this algorithm we should firstly choose the necessary error of the approximation of histogram of relative frequencies η and then assign the PDF to each state of the 4-state modulating Markov chain to yield (10). Note that the time the algorithm needs to find a suitable solution depends on the error η.

7.5 Modeling Results

We have applied both algorithms [81, 85] to all traces from the MPEG trace archive [78] and found that it matches both the histogram of relative frequencies of GoP sizes and the empirical ACF fairly well. In order to compare the model with empirical data we generated exactly the same number of *I*-, *P*-, *B*- and GoP sizes as the empirical traces. Here we present a discussion of comparison studies between both models and the *"Star Wars"* trace given in [81, 85].

In order to compare the distribution function of the models with corresponding histograms of relative frequencies (*I*-, *P*- and *B*-frame sizes for frame level and GoP sizes for GoP level) we have performed a chi-square statistical test. The test has shown that the statistical data of GoP sizes belong to the PDF of the D-BMAP model given a significance level of 0,05 and the statistical data of *I*-, *P*- and *B*-frame sizes belong to corresponding distribution functions of the GBAR(1) model given a significance level of 0,1.

Considering the ACF behavior we note that of lags up to 50-100 the empirical ACF and the ACF of the GBAR model are close to each other. Later, the ACF of the model quickly decays to zero. With the increasing of *p* parameter the correlation has a strong trend to rise for all lags. Thus, the model overestimates the ACF up to lag 80 and, consequently, is not able to give a good approximation for larger lags. At the frame level the D-BMAP model approximates the behavior of the empirical ACF of GoP sizes well for any lag.

One of the major shortcomings of the GBAR(1) model stems from the fact that the real data trace contains several frames with a very high number of bytes. The model can approximate this property in case of small values of p parameter, which will lead to underestimation of the ACF for large lags. There is a trade-off between two properties of the model: the distribution of frame sizes and the ACF. One possible solution is the careful choice of p values.

7.6 Conclusions

In this section we have considered the modeling of an MPEG-1 video elementary stream at the output of the codec. We propose two MPEG traffic models aimed at different MPEG logical levels: frame and group of pictures (GoP) levels.

On the frame level we proposed an extension of a modeling methodology proposed in [79] and [77]. We have used a two-step approach in order to model the video frames sequence. For the first step we approximate the I-frames generation process by a GBAR process. The second step consists of the approximation of frame sizes based on both the output of the GBAR process and intra-GoP correlations. The proposed algorithm provides a simple model of the MPEG source based on three cross-correlated processes. The algorithm captures well both the distribution of frame sizes and the ACF of empirical data. The GBAR model is fast, computationally efficient and captures well the SRD behavior of the ACF and the distribution of the frame sizes. It can be used in simulation studies where there is a need to generate MPEG traffic in real time.

The GBAR process needs only a few parameters, which can be estimated directly from the analysis of empirical data. This is a big advantage of the GBAR source model compared to other models of video traffic sources.

Our model at the GoP level is a refinement of the model originally proposed in [77]. The model of smoothed MPEG traffic at the GoP level from a single MPEG source is based on a special type of D-BMAP process. Based on the statistical analysis of MPEG traffic at the GoP level we limited the state space of the modulating Markov chain of the D-BMAP process. The limited state space allows us to decrease the complexity of the algorithm while the necessary accuracy of the approximation is retained. The D-BMAP model captures both the histogram of relative frequencies and the empirical ACF of a single smoothed MPEG traffic source. The model is useful for both analytical evaluation of queuing systems and simulation studies.

8 An Open Architecture for Diffserv-Enabled MPLS Networks

8.1 Introduction

The Internet has quickly evolved into a very critical communications infrastructure, supporting significant economic, educational and social activities. Simultaneously, the delivery of Internet communication services has become very competitive and end-users are demanding very high quality services from their service providers. Consequently, performance optimization of large scale IP networks, especially public Internet backbones, has become an important problem. This problem is addressed by traffic engineering, which encompasses the application of technology and scientific principles to

the measurement, characterization, modeling and control of Internet traffic. Enhancing the performance of an operational network, at both the traffic and resource levels, are major objectives of Internet traffic engineering; this is accomplished by addressing traffic oriented performance requirements, while utilizing network resources economically and reliably. Traffic engineering deals essentially with the selection of optimal paths that different flows should follow in order to optimize resource utilization and satisfy each flow's requirements. Historically, effective traffic engineering has been difficult to achieve in public IP networks, due to the limitations of legacy IGPs, which are adaptations of shortest path algorithms where costs are based on link metrics. This can easily lead to unfavorable scenarios in which some links become congested while others remain lightly loaded [88].

The development and introduction of Multi-Protocol Label Switching (MPLS) [89] has opened new possibilities to address some of the limitations of IP systems concerning traffic engineering. Although MPLS is a relatively simple technology (based on the classical label swapping paradigm), it enables the introduction of traffic engineering function in IP networks because of its support of explicit LSPs, which allow constraint-based routing (CBR) to be implemented efficiently. For this reason, MPLS currently appears as the best choice to implement traffic engineering in IP networks [90].

It is clear, however, that in modern IP networks, the need for supporting flows with different QoS requirements would demand additional mechanisms, such as those that perform policing on the incoming traffic, classify packets in different service classes and assure the flows the required quality of service. The introduction of appropriate packet scheduling disciplines and of architectures for differentiating services, such as Diffserv [91], can be used for this purpose. When MPLS is combined with Diffserv and explicit routing, we have a powerful architecture which allows both traffic engineering and quality of service provisioning. The interoperability between MPLS and Diffserv has already been the subject of studies from the Internet community and the resulting architecture has been defined in RFC 3270 [92].

Believing in the benefits of a Diffserv-enabled MPLS architecture, this chapter presents an experimental approach to the study of the interoperability between the Diffserv and MPLS paradigms. Few implementations currently exist of both architectures and most of them represent proprietary solutions, with device-specific contaminations. With respect to Diffserv, two different open source implementations have been considered, based respectively on the FreeBSD and Linux operating systems. The former is the ALTQ [93] package developed at the Sony Research Laboratories in Japan; the latter is the Traffic Control (TC) [94][95] module. Such modules basically make the fundamental traffic control components available, which are needed in order to realize the Diffserv paradigm: classifiers, schedulers, conditioners, markers, shapers, etc. As far as MPLS is concerned, a package running under Linux has been utilized [96]. In order to evaluate the performance overhead due to pushing/popping the MPLS label measurements have been performed, explained in Section 8.2.

The testbed set up at the University of Naples makes use of the Linux implementations of both Diffserv and MPLS to analyze four different scenarios (best effort, Diffserv, MPLS and Diffserv over MPLS), in order to appreciate the efficiency of Diffserv in traffic differentiation and to evaluate the interoperability between MPLS and Diff-

serv. Section 8.3 describes the experimental testbed, while Section 8.4 shows the results obtained from experimentations and draws a comparison among the four different scenarios.

We point out that the trials described in this chapter are carried out by statically configuring each router, while a dynamic configuration is required to serve each incoming service request as it arrives (it is necessary to update policers and filters, create a new LSP, etc.). Therefore, the next step is to develop an infrastructure for the dynamic configuration of routers. Some work in this direction has already been done, as described in Section 8.5. Conclusions and directions of future work are provided in Section 8.6.

8.2 MPLS: Protocol Overhead

This section presents an analysis of the overhead introduced by the MPLS encapsulation; for this purpose, two Linux PCs were put on a hub and a client-server application was used to measure the round-trip time of packets.

We present here only the results obtained generating UDP traffic; more details can be found in [97]. Figure 20 compares the mean RTT (in microseconds) for each of the 10 trials carried out in the simple IP case and in the MPLS case, for a packet length of 1000 bytes. The first thing to note is that the mean RTT in the MPLS case is always greater than the one in the IP case; this implies that the insertion and the extraction of a label introduces a certain overhead. Now we want to focus on how much greater this overhead is and how it changes with varying the packet size. The second graph of Figure 20 shows the percentage difference between the mean RTTs (calculated over all the 10 trials) in the two cases, for three packet lengths (10, 100 and 1000 bytes). This graph shows that the percentage difference decreases when the packets size increases. The reason for this behavior is that by increasing the packet size, the transmission time (protocol independent) increases while the processing time (protocol dependent) remains the same. Thus the protocol dependent component is less and less important and the difference between the two RTT values tends to diminish.

These measures show that the introduction of MPLS comes with a certain cost, and thus this effect should be taken into account when we use QoS mechanisms with MPLS. Therefore purpose, in the next sections we examine the performance relative to the four configurations (best-effort, Diffserv, MPLS and Diffserv-MPLS); first, we need to describe the experimental testbed.

8.3 The Experimental Testbed

In this section we present a test platform developed at the University of Naples with the intent of gaining experience from actual trials and experimentations. Such a testbed, shown in Figure 21, consists of Linux routers and closely mimics (apart from the scale factor) an actual internetworking scenario. It is built of a number of interconnected LAN subnets, each realized by means of one or more cross-connections between pairs of routers. For these tests, routers A, B and C represent the QoS-enabled IP infrastructure whose performance is under evaluation; host A acts as a traffic generator and also as a sink. The direct connection from host A to host B was added in order to allow a precise evaluation of transmission time for the packets: host A just sends data from

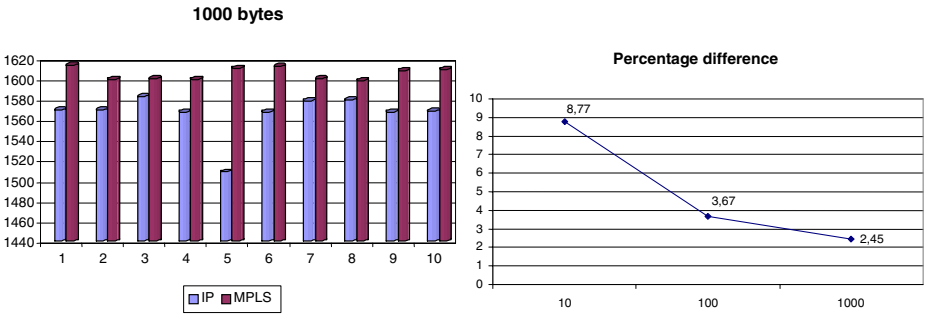


Fig. 20. Experimental results: round-trip-times.

one of its interfaces to the other, data flows first arrive to host B (thus crossing the QoS backbone) and then directly back to host A. Before going into the detailed performance analysis for the four different configurations (Best Effort, Diffserv, MPLS and Diffserv over MPLS) we explain the traffic pattern adopted for all of the experiments. We used Mtools, a traffic generator developed at University of Naples [98][99], which lets one choose packet dimensions, average transmission rates and traffic profiles (either deterministic or Poisson). The Diffserv Code Point (DSCP) of the generated packets can also be set. An interesting feature of Mtools is the capability to reproduce the same realization of the packet generation random process, by setting the same generating seed for different trials. We fed the network with four Poisson flows (generated by host A), each requiring a different Per Hop Behavior (EF, DE, AF11, AF12). The overall bit rate of the traffic injected into the network is about 3 Mbps. In order to appreciate the impact of scheduling (and, in the case of Diffserv, also policing) on the network, we configured all of the routers to use a Class Based Queuing (CBQ) scheduler, with a maximum available bandwidth of 2.8 Mbps. Since the injected traffic is more than the network can manage, packet losses will be experienced and we expect them to be uniformly distributed across the best effort and plain MPLS (i.e. MPLS with just one LSP carrying all the traffic) scenarios. More details on the generated traffic can be found in [100].

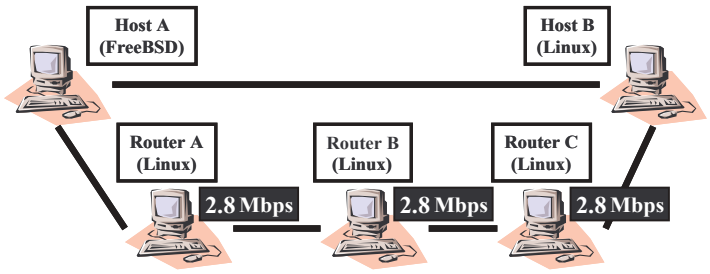


Fig. 21. Testbed used for the trials.

In the best effort case, we have just one CBQ class for all of the traffic flows (which are thus scheduled without taking into consideration the DSCP inside the IP header). All the available output bandwidth has been assigned to this class.

In the Diffserv scenario, instead, of the available 2.8 Mbps, 0.8 are assigned to the EF class, 1.0 to AF1 and 1.0 to default traffic (DE). Expedited Forwarding is served in a first in first out (FIFO) fashion, while the Assured Forwarding and default behavior queues are managed, respectively, with a Generalized Random Early Discard (GRED) and a Random Early Discard (RED) algorithm. Referring to [100] for further details, we want to point out that the ingress Diffserv edge router (router A), unlike the others (routers B and C), has been configured with a policer, whose function is to either drop or re-mark out-of-profile packets.

In a pure MPLS network, packets are assigned to the various Label Switched Paths (LSPs) based on information such as sender's and receiver's IP addresses. More specific fields, such as the DSCP (i.e the Type of Service byte of the IP header), are completely ignored. In this scenario we will thus rely on a single LSP to carry all of the traffic flows that enter the MPLS cloud. This means that all packets will be marked with the same label and will experience the same treatment.

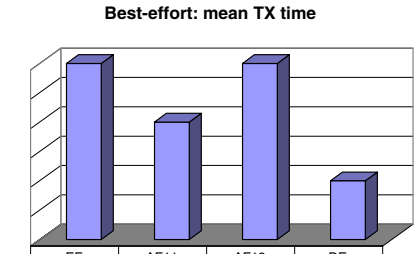
In contrast, the Diffserv over MPLS scenario provides for MPLS support to Diffserv, that is to say packets are forwarded via MPLS label swapping, but different packet flows (as identified by the DSCP code) are treated in a different fashion. This is achieved by inserting additional information into the MPLS header. Here we will exploit one LSP for EF and DE traffic and a different one for the two AF1 flavors. In the first case, information about the Per Hop Behavior will be encoded in the EXP bits of the MPLS header, thus creating a so-called E-LSP [92]; on the other hand, for the Assured Forwarding flows (which have to be scheduled in the same fashion) the EXP bits carry information related to the drop precedence level (L-LSP). For this trial, again, router A has to perform policing at its ingress interface.

8.4 Experimental Results

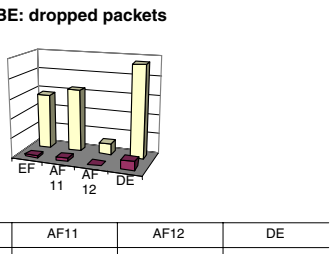
In this section we will show, comment and compare the experimental results we obtained for the four aforementioned scenarios. As a preliminary consideration, please notice that transmission time is a useful indicator when evaluating performance, since it contains information related to two fundamental aspects: the time needed to process packets and the time spent waiting inside queues.

Best Effort Scenario We will start by analyzing the results obtained with the Best Effort network setup. Since no differentiation is provided in this case, we expect that all the flows receive more or less the same treatment. This is what actually happens, as witnessed by the first graph of Figure 22 and the table below, which show the mean transmission delay for every flow. From the second graph, notice that packet losses are directly proportional to the different number of sent packets for the flows.

Diffserv Scenario Let us now switch to Diffserv. What we expect now is that the requirements we specified for the single flows are actually met: EF packets should be



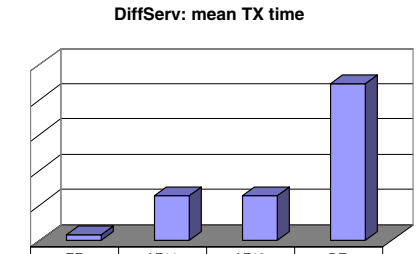
	EF	AF11	AF12	DE
BE	0,251	0,25	0,251	0,249



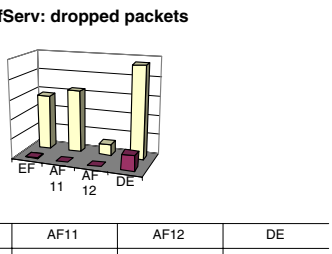
	EF	AF11	AF12	DE
dropped	99	185	3	453
sent	2807	3204	574	4724

Fig. 22. Best effort scenario

forwarded much faster than the others, AF packets should be reliably delivered, while DE packets should be treated in a Best Effort fashion. The first graph of Figure 23 reports mean transmission times, while the second shows the number of dropped packets in the Diffserv case. The first comment is that packets belonging to different flows are definitely treated in a different manner: transmission delays vary from one class to the other and this was exactly what we envisaged, since Diffserv is nothing but a strategy to differentiate packets based on their specific requirements. EF packets are those that endure the smallest delay; AF packets, in turn, experience a reliable delivery (as witnessed by the zero packets lost for both AF11 and AF12 classes). Finally, the DE class is the one which suffers from the highest delay and the greatest packet losses.



	EF	AF11	AF12	DE
DS	0,015	0,127	0,127	0,444

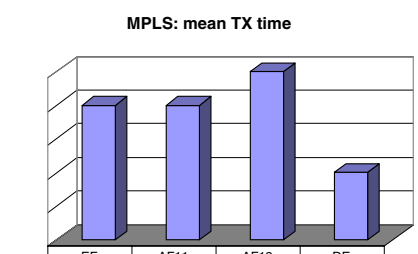


	EF	AF11	AF12	DE
dropped	11	0	0	769
sent	2807	3204	574	4725

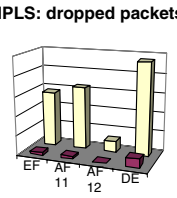
Fig. 23. Diffserv scenario

MPLS Scenario In the MPLS scenario all the flows should be treated the same way, since no differentiation mechanism has been set up and just one LSP is in place. Figure 24 shows the mean transmission delay and the number of dropped packets for the MPLS configuration. The plots are similar to those we showed for the Best Effort case. With the enforced network setup the actual bottleneck is definitely represented by the output interface's bandwidth, hence the contribution due to packet processing is negligible

when compared to the others: that is why we were not able to appreciate the performance difference between plain IP and MPLS we observed and measured in Section 8.2.



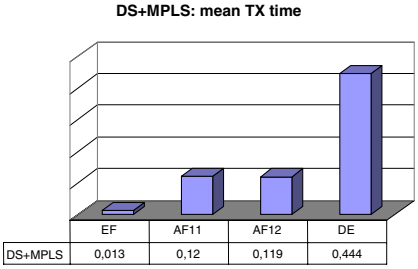
	EF	AF11	AF12	DE
MPLS	0,251	0,251	0,252	0,249



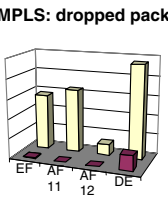
	EF	AF11	AF12	DE
dropped	141	152	2	445
sent	2808	3204	574	4725

Fig. 24. MPLS scenario

Diffserv over MPLS Scenario Let us finally take a look at the results obtained with the last configuration, where Diffserv is running over an MPLS backbone. Figure 25 shows how the presence of a Diffserv infrastructure meets the requirements for both EF and AF flows. By comparing the graphs in Figure 23 with those in Figure 25 we notice that the traffic profiles are almost the same in the two scenarios. Though MPLS adds little to network performance in the context described (i.e. one in which the routing tables dimensions are pretty small), its major benefit is that it can take advantage of constraint-based routing. Stated in different terms, a comparison based solely on the delay performance figures is not fair, since a thorough analysis cannot avoid considering the fact that MPLS enables traffic engineering, by evading traditional (e.g. shortest path) forwarding, in favor of fully customized routing paradigms.



	EF	AF11	AF12	DE
DS+MPLS	0,013	0,12	0,119	0,444



	EF	AF11	AF12	DE
dropped	10	0	0	774
sent	2807	3204	574	4724

Fig. 25. DS+MPLS scenario

8.5 Dynamic Network Configuration

As already stated, the network configurations of the four scenarios depicted in the previous section are static and manually enforced. In order to develop a general framework for the effective negotiation and delivery of services with quality assurance, it is necessary to realize an infrastructure for the dynamic and automatic configuration of the network elements.

In this section, we briefly outline the building blocks of such an architecture. The core of this architecture is a centralized element called *Network Controller* (NC), whose task is to configure network elements so as to satisfy the conditions included in the SLSs. These represent the contracts stipulated with the users (SLAs) in technical terms. The NC can make use of two traffic engineering algorithms (one online and the other offline) in order to perform its task of admitting service requests and determining which paths the packets should follow. Once the NC has taken its decisions, it must communicate them to the network elements, so they can be enforced; these communications take place by means of the COPS (Common Open Policy Service) protocol. COPS defines the set of messages required to send the so-called *policies*, that is the actions to perform when certain conditions occur.

The set of policies that constitutes the current network configuration is stored in an LDAP directory; this enables information to be stored on a non-volatile media and also provides a mean for the quick recovery of the configuration whenever a network element should fail and loose its configuration.

The part of the NC which is responsible of the transmission of policies is called, in COPS terminology the PDP (Policy Decision Point). The PDP asks the LDAP server for the policies and sends them to the PEPs (Policy Enforcement Points), which are processes running on network elements. Each PEP, in its turn, passes these policies to the Device Controller (DC), which translates them into a set of commands to actually configure the device. The DC is subdivided into three parts: the first one that installs filters, markers, policers and packet scheduling disciplines; the second one that deals with the creation of E-LSP [92], and the last one that is responsible of mapping traffic on the appropriate LSP.

8.6 Conclusions and Future Work

A number of experiments aimed at investigating network performance in the presence of either Diffserv, or MPLS, or a hybrid Diffserv/MPLS infrastructure, have been presented in this section.

For future work, the challenge is that of dynamically controlling such advanced networks, by applying a policy-based management paradigm to them. We will look at the realization of a complex architecture for Service Assurance, capable to use the advantages of the two technologies. A traffic-engineered Diffserv environment (TRADE) is the final target of this specific research: the major ingredients behind it are a Diffserv-capable network on top of a traffic-engineered MPLS backbone.

9 Wide Area Measurements of Voice over IP Quality

It is well known that the users of real time voice services are sensitive and susceptible to variable audio quality. If the quality deteriorates below an acceptable level or is too variable, users often abandon their calls. Since the Internet is increasingly being used to carry real time voice traffic, the quality provided has become, and will remain an important issue. The aim of this work is therefore to report the current quality of voice communication at end-points on the Internet.

It is intended that the results of this work will be useful to many different communities involved with real time voice communication. We now list some potential groups to whom this work might be relevant. Firstly end users, who can use the (past) measurements to determine which calling destinations will provide sufficient quality. When deemed insufficient they can possibly take preventative measures such as adding robustness, for example in the form of forward error correction (FEC) to their conversations. Operators can use findings such as these to motivate upgrading links or adding QoS mechanisms where poor quality is being reported. Network regulators can use this kind of work to verify that the quality level that was agreed upon, has indeed been deployed. Speech coder designers can utilise the data for new classes of codecs. Of current interest are designs which yield good quality in the face of bursty packet loss. Finally, researchers could use the data to compare with their own measurements, to investigate trends for example "Is the quality of real time audio communication improving or deteriorating?".

The structure of the measurement work is as follows: Section 9.1 begins with a brief background on the quality measures we have used in this work namely, loss, delay and jitter. Following on from the quality measures, Section 9.2 gives a description of the methodology used to ascertain the quality. In Section 9.6 the results are presented, and due to space considerations, we condense the results into one table showing the delay, loss and jitter values for each of the paths we measured. In Section 9.7 the related work is presented, comparing results we obtained with other researchers' work. This is considered important as it indicates whether quality has improved or deteriorated since the related work was conducted. Section 9.8 rounds off with some conclusions and includes a reference to where the measurement data can be found.

9.1 What Do We Mean by Voice over IP Quality?

Ultimately, users judge the quality of voice transmissions. Organisations such as ETSI, ITU, TIA, RCR plus many others have detailed mechanisms to assess voice quality. Speech coding is usually the focus of these types of organisations. Quality assessment in these forums involves replaying coded speech to both experienced and novice listeners, and then asking them to rate or rank the perceived quality. Judging the quality of voice data that has been transmitted across a wide area network is more difficult. The network inflicts its own characteristics on the quality of the voice stream. By measuring the delay, jitter and loss of a data stream at a receiver, we can provide some indication of how the network disrupts the original speech. Additionally we can say how suitable the network (more explictly the connections) is for real time voice communication. It is

important to state however, we cannot reproduce controlled and exact quality measures as is done with speech coder evaluation.

The quality of VoIP sessions can often be quantified by network delay, packet loss and packet jitter. We emphasise that these three quantities are the major contributors to the perceived quality as far as the network is concerned. The G.114 ITU standard states the end-to-end one way delay should not exceed 150ms [101]. Delays over this value adversely affect the quality of the conversation. In an alternative study, Cole and Rosenbluth state that users perceive a linear degradation in the quality up to 177ms [102]. Above this figure the degradation is also linear, although markedly worse. As far as the packet loss is concerned, using simple speech coding, e.g. A-law or μ-law, tests have shown that the mean packet loss should not exceed 10% with packet loss concealment and 1% without. Packet loss concealment is a technique for reducing quality degradation due to lost packets. Typically the missing samples from lost packets are generated from an interpolation of the samples that were received. Therefore glitches due to lost packets do not seriously affect the perceived quality. This is evident from the substantially different tolerance to loss, with and without packet loss concealment. Note that a loss rate such as this does not say anything about the distribution of the losses. As far as the authors are aware of, no results exist that state how jitter alone affects the quality of voice communication. When de-jittering mechanisms are employed, the network jitter is typically transferred into application *delay*. The application must hold back a sufficient number of packets in order to ensure smooth, uninterrupted playback of speech. To summarise, we refer to the quality as a combination of delay, jitter and loss. It is important to mention that we do not explicitly state how these values should be combined. The ITU E-model [103] is one approach but others exist. Therefore we refer the interested reader to the references in this section as well as [104] and [105].

9.2 Simulating and Measuring Voice over IP Sessions

Our method to measure VoIP quality is to send pre-recorded calls between globally distributed sites. Through the modification of our own VoIP tool, Sicsophone, the intervening network paths are probed by a 70 second pre-recorded 'test signal'. The goal of this work summarises the state of the signal after traversing several different network paths. Incidentally, we do not include the signalling phase (i.e. establishing communication with the remote device) in these measurements, rather we concentrate solely on the data transfer quality.

Nine sites have been carefully chosen with large variations in hops, geographic distances, time zones and connectivity to obtain a diverse selection of distributed sites. One limitation of the available sites was that they were all located at academic institutions, which are typically associated with well provisioned networks. Their locations are shown in the map of Figure 26. The sites were connected as a full mesh allowing us, in theory, to measure the quality of 72 different Internet paths. In practice, some of the combinations were not possible due to certain ports being blocked, thus preventing the audio to be sent between some sites. There were four such cases. Bi-directional sessions were scheduled on an hourly basis between any two end systems. Calls were only transferred once per hour due to load considerations on remote machines.

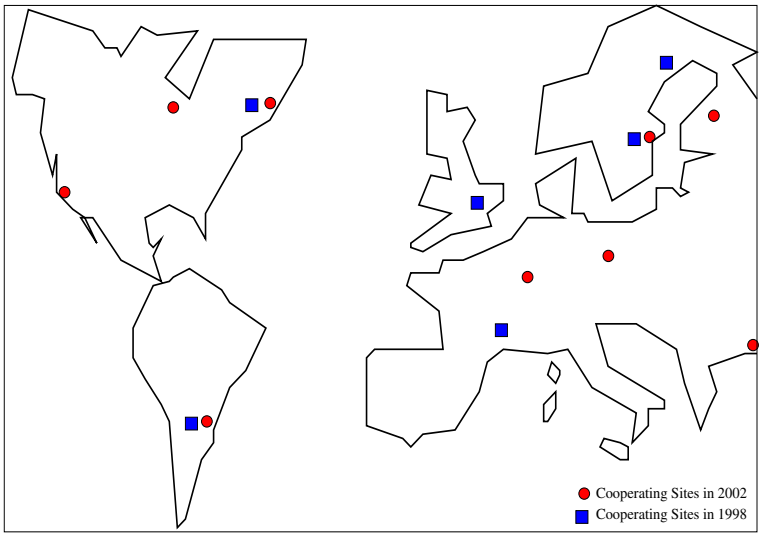


Fig. 26. The nine sites used in 2002 are shown with circles. The six depicted with squares show those that were available to us in 1998, three remained unchanged during the past four years.

In Table 3 we list the characteristics of the call used to probe the Internet paths indicated on the map (squares). Their locations, separation in hops and time zones are given in the results Section. As stated, the call is essentially a fixed length PCM coded file which can be sent between the sites, the length of the call and the payload size were arbitrarily chosen. Over a 15 week period we gathered just over 18,000 recorded sessions. We attempted to ensure an even number of measurements per site. In total nearly 33 million individual packets were received during this work.

9.3 A Networking Definition of Delay

We refer to the delay as the *one way network* delay. One way delay is important in voice communication, particularly if it is not equal in each direction. Measuring the one way delay of network connections without the use of synchronised clocks is not a simple task. Hence many methods rely on round-trip measurements and halve the result, estimating the one way delay. We measured the network delay using the RTCP protocol which is part of the RTP standard [106]. A brief description follows. At given intervals the sender transmits a 'report' containing the time the report was sent. On reception of this report the receiver records the current time. Therefore two times are recorded within the report. When returning the report to the sender, the receiver subtracts the time it put in the report, therefore accounting for the time it held the report. Using this information the sender can calculate the round-trip delay and importantly, discount the time spent processing the report at the receiver. This can be done in both directions to see if any significant differences exist.

Table 3. The top half of the table gives details of the call used to measure the quality of links between the sites. The lower half provides information about the data which we gathered.

Test 'signal'	
Call duration	70 seconds
Payload size	160 bytes
Packetisation time (ms)	20ms
Data rate	64kbits/sec
With silence suppression	2043 packets
Without silence suppression	3653 packets
Coding	8 bit PCM
Recorded call size	584480 bytes
Obtained data	
Number of hosts used (2002)	9
Number of traces obtained	18054
Number of data packets	32,771,021
Total data size (compressed)	411 Megabytes
Measurement duration	15 weeks

The delay quoted in this work should be regarded as the network delay. The contribution of the operating systems in this work is minimal and the application is not involved (no jitter removal was implemented). We verified the application is not involved by comparing the RTCP results with regular ICMP echo/requests. The results were very similar, indicating the delay added by the operating systems was negligible. Another reason for not including the delay contributed by the end systems is it can vary from one operating system to another, and it also depends on how the VoIP application is implemented. The delay added by an end system can vary from 20ms up to 1000ms, irrespective of the stream characteristics.

9.4 Jitter - An IETF Definition

Jitter is the statistical variance of the packet interarrival time. In RFC 1889 the jitter is defined to be the mean deviation (the smoothed absolute value) of the packet spacing change between the sender and the receiver [106]. Sicsophone sends packets of identical size at constant intervals, which implies that $S_j - S_i$ (the sending times of two consecutive packets) is constant. The packet spacing, denoted D, is used to calculate the interarrival jitter. The interarrival jitter should be calculated continuously as each packet i is received. For one particular packet, the interarrival jitter J_{i-1} for the previous packet $i-1$ is calculated thus:

$$J_i = J_{i-1} + (|D(i-1,i)| - J_{i-1})/16.$$

According to the RFC "The gain parameter 1/16 gives a good noise reduction ratio while maintaining a reasonable rate of convergence". As stated earlier buffering due to jitter adds to the delay of the application. The jitter therefore is not visible in the delay results we present. This value can also be implementation dependent as the time

needed for de-jittering depends on how the original time spacing of the packets should be restored. Many different techniques exist, so we do not tie the measurements to a particular de-jittering scheme, we do give the jitter for each connection though. As a simple example, if a single packet buffer is employed it would result in an extra 20ms (the packetisation time) being added to the total delay. Note that packets arriving with a spacing greater than 20ms should be discarded by the application as being too late for replay. Multiples of 20ms can thus be allocated for every packet held before playout in this simple example.

9.5 Counting Ones Losses in the Network

We calculate the lost packets as defined in RFC 1889. The number of lost packets is the expected number of packets subtracted from the number actually received. The loss is calculated using expected values so as to allow more significance for the number of packets received, for example, 20 packets lost from 100 packets sent has a higher significance than 1 lost from 5 sent. In this work losses *do not* include those incurred by late arrivals, as knowledge of the buffer playout algorithm is needed, therefore our values are solely the network loss. Detailed analysis of the loss patterns is not given, we simply state the percentages of single, double and triplicate losses.

9.6 Results

The results of 15 weeks of measurements are condensed into Figure 27.

The table should be interpreted as an 11x11 matrix. The locations listed horizontally across the top of the table are the locations configured as receivers, and when listed vertically are configured as senders. The values in the rightmost column and bottom row are the statistical means for all the connections *from* the host in the same row and *to* the host in the same column respectively. For example the last column of the first row (directly under the 'Mean' heading) is the average delay to all destinations from Massachusetts, in this case 112.8ms.

Each cell includes the delay, jitter, loss, number of hops and the time difference indicated by the letters D, J, L, H and T for each of the 72 connections. The units for each quantity is the delay in milliseconds, the jitter in milliseconds, the loss in percentage, the hops as reported by traceroute and the time differences in hours. A '+' indicates that the local time from a site is ahead of the one in the corresponding cell and behind for '-'. The values in parentheses are the standard deviations. NA signifies "Not Available" for this particular combination of hosts. The bottom rightmost cell contains the mean for all 18054 calls made, both to and from all nine hosts.

The most general observation is the quality of the paths is generally good. The average delay is just below the ITU's G.114 recommendation for end-to-end delay. Nevertheless at 136ms it does not leave much time for the end systems to encode/decode and replay the voice stream. A small buffer would absorb the 4.1ms jitter and a loss rate of 1.8% is more than acceptable with PCM coding [104].

There are two clear groupings from these results: those within the EU and the US and those outside. The connections in Europe and the United States, and between them, are very good. The average delay between the US/EU hosts is 105ms, the jitter is 3.76ms

receiver / sender	Massachusetts	Michigan	California	Belgium	Finland	Sweden	Germany	Turkey	Argentina	Mean
Massachusetts	*	D:38.0 (17.1) J:2.4 (1.7) L:0.1 (0.6) H:14 (+1) T:0	D:54.2 (15.8) J:2.4 (1.8) L:0.1 (0.9) H:19 T:-3	D:67.1 (15.5) J:3.6 (1.5) L:0.1 (0.8) H:11 T:+6	D:97.1 (2.6) J:2.5 (1.5) L:0.1 (0.8) H:15 T:+7	D:99.5 (8.5) J:3.2 (1.7) L:0.04 (0.2) H:21 T:+6	D:58.4 (5.0) J:4.5 (1.4) L:0.0 (0.0) H:17 (+3) T:+6	D:388.2 (43.2) J:10.4 (4.9) L:4.9 (4.7) H:20 T:+7	D:99.7 (4.9) J:19.9 (8.4) L:8.9 (7.2) H:25 T:+1	D:112.8 J:6.1 L:1.2 H:17
Michigan	D:36.4 (15.4) J:4.7 (0.8) L:0.0(0.2) H:14 (+1) T:0	*	D:40.4 (4.5) J:4.4 (0.8) L:0.2 (1.1) H:20 (+1) T:-3	D:63.5 (4.2) J:4.3 (0.7) L:0.0 (0.1) H:11 T:+6	D:88.2 (8.0) J:4.1 (0.7) L:0.1 (1.1) H:17 T:+7	D:86.7 (4.7) J:5.2 (0.6) L:0.1 (2.2) H:23 T:+6	D:63.6 (8.2) J:7.3 (1.9) L:0.2 (0.9) H:16 (+1) T:6	D:358.9 (44.9) J:5.6 (1.7) L:3.0 (1.9) H:20 T:7	D:112.1 (10.6) J:18.7 (7.9) L:6.5 (7.0) H:25 T:+1	D:106.2 J:6.8 L:1.3 H:18
California	D:54.5 (16.7) J:2.0 (1.0) L:0.1 (0.36) H:18 (+1) T:+3	D:40.6 (5.1) J:1.2 (0.6) L:0.1 (1.9) H:21 T:+3	*	D:81.0 (2.2) J:1.6 (0.8) L:0.2 (0.8) H:20 T:+9	D:106.0 (3.0) J:1.4 (0.8) L:0.6 (1.4) H:25 (+1) T:+10	D:108.0 (2.4) J:2.1 (0.9) L:0.2 (0.3) H:30 (+2) T:+9	D:81.5 (1.8) J:4.9 (1.5) L:2.8 (3.0) H:23 T:+9	D:386.9 (60.5) J:5.3 (1.7) L:4.4 (2.4) H:23 T:+10	D:123.9 (12.4) J:18.1 (9.9) L:8.9 (8.2) H:25 T:+4	D:122.2 J:4.6 L:2.2 H:23
Belgium	D:65.2 (10.1) J:1.6 (0.6) L:0.0 (0.0) H:16 T:-6	D:63.4 (3.3) J:0.6 (0.1) L:0.0 (0.0) H:17 T:-6	D:84.0 (1.3) J:0.9 (0.8) L:1.2 (1.0) H:23 T:-9	*	D:31.3 (0.6) J:0.9 (0.5) L:0.0 (0.0) H:17 T:+1	D:33.4 (0.2) J:1.6 (0.9) L:0.0(0.0) H:22 T:0	D:16.6 (10.4) J:3.4 (1.5) L:0.21 (0.7) H:13 T:0	D:341.1 (24.7) J:6.9 (2.0) L:3.8 (2.7) H:16 (+2) T:+1	D:136.5 (7.1) J:NA L:NA H:19 T:-5	D:96.4 J:2.0 L:0.6 H:17
Finland	D:97.8 (4.2) J:1.7 (0.8) L:0.0 (0.1) H:15 (+1) T:-7	D:86.8 (1.9) J:1.1 (0.6) L:0.0 (0.3) H:17 (+1) T:-7	D:109.9 (4.7) J:1.4 (0.8) L:0.7 (1.4) H:24 (+2) T:-10	D:30.7 (0.3) J:1.4 (0.6) L:0.1 (0.3) H:16 T:-1	*	D:13.6 (1.0) J:1.9 (0.9) L:0.0 (0.0) H:20 T:-1	D:26.8 (7.3) J:3.9 (1.1) L:0.0(0.0) H:20 (+1) T:-1	D:321.2 (39.3) J:3.4 (1.7) L:3.2 (1.7) H:17 (+2) T:0	D:161.5 (12.2) J:17.4 (8.2) L:7.5 (6.5) H:19 T:-6	D:106.3 J:4.1 L:1.4 H:18
Sweden	D:99.3 (8.8) J:3.0 (1.9) L:0.0 (0.0) H:22 (+1) T:-6	D:84.9(1.9) J:2.5 (2.0) L:0.03 (0.4) H:25 T:-6	D:105.6 (2.1) J:3.2 (1.96) L:0.1 (0.1) H:30 T:-9	D:33.3 (0.4) J:2.8 (1.6) L:0.1 (0.3) H:24 T:0	D:13.5 (0.5) J:2.4 (1.8) L:0.0 (0.01) H:21 T:+1	*	D:29.8 (12.8) J:4.8 (2.5) L:0.0 (0.0) H:25 T:0	D:322.2 (30.3) J:3.2 (1.49) L:2.9 (1.0) H:26 T:+1	D:165.6 (17.9) J:NA L:NA H:41 T:-5	D:107.8 J:2.8 L:0.4 H:26
Germany	D:63.5 (9.6) J:1.72 (0.7) L:0.0(0.0) H:15 T:-6	D:60.4 (0.5) J:0.7 (0.3) L:0.0 (0.0) H:16 T:-6	D:84.4 (1.0) J:1.8 (0.7) L:2.5 (1.9) H:22 T:-9	D:11.1 (0.2) J:0.8 (0.3) L:0.0 (0.0) H:12 T:0	D:27.8 (7.3) J:1.0 (0.5) L:0.0 (0.0) H:17 T:+1	D:29.2 (7.6) J:1.5 (0.6) L:0.0 (0.0) H:22 T:0	*	D:300.7 (39.7) J:4.8 (2.1) L:3.7 (2.5) H:16 T:+1	D:149.8 (15.6) J:NA L:NA H:18 T:-5	D:90.9 J:1.6 L:0.8 H:17
Turkey	D:379.1 (47.1) J:8.6 (0.7) L:8.1 (2.8) H:18 (+1) T:-7	D:387.9 (35.5) J:8.9 (1.2) L:8.0 (2.9) H:20 T:-7	D:410.9 (43.9) J:8.8 (2.5) L:7.6 (6.8) H:19 T:-10	D:330.2 (28.6) J:9.2 (2.0) L:7.10 (4.0) H:17 T:-1	D:318.9 (42.4) J:8.8 (0.6) L:7.8 (2.7) H:19 T:0	D:311.1 (8.3) J:9.1 (0.7) L:8.4 (3.1) H:25 T:-1	D:378.2 (49.3) J:10.7 (1.2) L:8.0 (3.1) H:16 T:-1	*	D:490.8 (26.0) J:NA L:NA H:18 T:-6	D:375.9 J:8.0 L:6.9 H:19
Argentina	D:117.0 (30.8) J:4.2 (2.0) L:0.5 (1.4) H:NA T:-1	D:146.7 (44.2) J:4.3 (2.3) L:0.5 (1.5) H:NA T:-1	D:152.0 (47.8) J:3.1 (2.4) L:0.6 (1.8) H:NA T:-4	D:NA J:4.2 (2.0) L:0.5 (1.4) H:NA T:+5	D:164.1 (27.2) J:3.9(2.2) L:0.5 (1.4) H:NA T:+6	D:160.9 (47.7) J:2.9 (0.8) L:0.0 (0.1) H:NA T:+5	D:180.5 (50.5) J:4.7 (1.5) L:0.1 (0.1) H:NA T:+5	D:NA J:6.0(1.2) L:5.8 (3.0) H:NA T:+6	*	D:115.2 J:4.2 L:1.1 H:NA
Mean	D:114.1 J:3.4 L:1.1 H:14	D:113.6 J:3.4 L:1.1 H:16	D:115.7 J:3.2 L:1.6 H:19	D:77.1 J:3.5 L:1.0 H:13	D:105.8 J:3.1 L:1.1 H:16	D:105.2 J:3.4 L:1.1 H:20	D:104.4 J:5.5 L:1.4 H:16	D:345.6 J:5.7 L:4.0 H:17	D:180.0 J:9.3 L:4.00 H:23	D:136.2 J:4.1 L:1.8 H:18

Fig. 27. A summary of 18000 VoIP sessions. The delay (D), jitter (J) and loss (L) for the nine sites. The delay and jitter are in milliseconds, the losses are in percentages. The number of hops and time zones (in hours) are also given. The means for each and all sites are given plus the standard deviations (in parentheses). An NA means 'Not Available'.

and the loss 1.16%. Those outside fair less well. The Turkish site suffers from large delays, which is not surprising as the Turkish research network is connected via a satellite link to Belgium (using the Geant network). The jitter and loss figures however are quite low, 5.7ms and 4%, respectively. The Argentinian site suffers from asymmetry problems. The quality when sending data to it is significantly worse than when receiving data from it. The delay is 1/3 higher, the jitter is more than twice that in the opposite direction and the loss is nearly four times higher than when sending to it. Unfortunately we could not perform a traceroute from the host in Buenos Aires due to not having super-user access, so we cannot say how the route contributed to these values.

We now turn our attention to results which are not related to any particular site. As far as loss is concerned the majority of losses are single losses. 78% of all the losses counted in all trace files were single losses whereas 13% were duplicate losses and only 4% triplicate losses. Generally the jitter is low relative to the delay of the link, approximately 3-4%. This is not totally unexpected as the loss rates are also low. With the exception of the Argentinian site, the sites did not exhibit large differences in asymmetry and were normally within 5% of each other in each direction. It is interesting to note that the number of hops could vary under the 15 week measurement period denoted by () in the hops field. Only very few ($< 0.001\%$) out of sequence packets were observed. Within [107] there are further results such as the effect of using silence suppression, differing payload sizes and daytime effects. In summary no significant differences were observed as these quantities were varied.

9.7 Related Work

Similar but less extensive measurements were performed by us in 1998 [108]. Only three of the hosts remain from four years ago so comparisons can only be made for these routes. An improvement, in the order of 5-10% has been observed for these routes. We should point out though, the number of sessions recorded four years ago numbered only tens per host, whereas on this occasion we performed hundreds of calls from each host. Bolot et. al. looked at consecutive loss for a FEC scheme [109]. They concluded that the number of consecutive losses is quite low and stated that most losses are one to five losses at 8am and between one to ten at 4pm. This is in broad agreement with the findings in this work. However we did not investigate the times during the day of the losses. Maxemchuk and Lo measured both loss and delay variation for intra-state connections within the USA and international links [110]. Their conclusion was that quality depends on the length of the connection and the time of day. We did not try different lengths of connection but saw much smaller variations (almost negligible) during a 24 hour cycle (see [107]). We attribute this to the small 64kbits per second VoIP session on well dimensioned academic networks. It is worth pointing out our loss rates were considerably less than Maxemchuk's (3-4%). Dong Lin had similar conclusions [111], stating that in fact even calls within the USA could suffer from large jitter delays. Her results on packet loss also agree with those in [109], which is interesting, as the measurements were taken some four years later.

9.8 Conclusions

We have presented the results of 15 weeks of voice over IP measurements consisting of over 18000 recorded VoIP sessions. We conclude that the quality of VoIP is very good and in most cases is over the requirements as stated in many speech quality recommendations. Remember that all of the sites were at academic institutions which is an important factor when interpreting these results, as most universities have well provisioned links, especially to other academic sites. Nevertheless, the loss, delay and jitter values are very low and comparing with previous measurements the quality trend is improving. We can only attribute this to more capacity and better managed networks than four years ago. However some caution is needed as the sample period was only 15 weeks, the bandwidth of the flows very small and only used once per hour. However we do have quite a large number of sample sessions.

Clearly VoIP is dependent on the IP network infra-structure and not only on the geographic distance. This can be seen in the differences between the Argentinian and Turkish hosts. We have found performing measurements on this scale is not an easy task. Different access mechanisms, firewalls, NATs and not having super-user permission complicates the work of obtaining measurements. We have made the data gathered as part of this effort publicly available for further investigation at http://www.sics.se/~ianm/COST263/cost263.html.

References

[1] Willinger, W., Paxson, V.: Where mathematics meets the internet. Notices of the American Mathematical Society **45** (1998) 961–970
[2] Park, K., Willinger, W.: Self-similar network traffic and performance evaluation. Wiley (2000)
[3] Padhye, J., Firoiu, V., Towsley, D., Kurose, J.: Modeling tcp throughput: a simple model and its empirical validation. In: Proceedings of SIGCOMM 98, ACM. (1998)
[4] Kleinrock, L.: Queuing Theory - Vol 2. Wiley (1976)
[5] Ben Fredj, S., Bonald, T., Proutire, A., Rgni, G., Roberts, J.: Statistical bandwidth sharing: a study of congestion at flow level. Proceedings of Sigcomm 2001, Computer Communication Review **31** (2001) 111–122
[6] Bonald, T., Roberts, J.: Congestion at flow level and the impact of user behaviour. Computer Networks, to appear (2003)
[7] Le Boudec, J.Y., Thiran, P.: A theory of deterministic queuing systems for the Internet. Volume 2050 of Lecture notes in computer science. Springer-Verlag (2001)
[8] Bonald, T., Proutire, A., Roberts, J.: Statistical performance guarantees for streaming flows using expedited forwarding. In: Proceedings of Infocom 2001. (2001) 1104–1112
[9] Gibbens, R., Kelly, F., Key, P.: A decision theoretic approach to call admission control. IEEE JSAC **13** (1995) 1104–1114
[10] Bonald, T., Oueslati, S., Roberts, J.: Ip traffic and qos control: towards a flow-aware architecture. In: Proceedings of World Telecom Conference, Paris (2002)
[11] Benameur, N., Ben Fredj, S., Delcoign, F., Oueslati-Boulahia, S., Roberts, J.: Integrated admission control for streaming and elastic flows. In: Proceedings of QofIS 2001, LNCS 2156, Springer (2001)
[12] Quadros, G., Monteiro, E., Boavida, F.: A qos metric for packet networks. In: Proceedings of SPIE International Symposium on Voice, Video, and Data Communications, Hynes Convention Center, Boston, Massachusetts, USA (1998)

[13] International Organization for Standardization: Information technology – quality of service: Framework. International Standard 13236 (1998)
[14] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.: Rfc 2475 – an architecture for differentiated service (1998)
[15] Quadros, G., Alves, A., Monteiro, E., Boavida, F.: An approach to support traffic classes in ip networks. In Crowcroft, J., Roberts, J., Smirnov, M.I., eds.: Lecture Notes in Computer Science. Volume 1922., Berlin, Germany, Springer-Verlag, Heidelberg (2000) 285–299
[16] Quadros, G., Alves, A., Monteiro, E., Boavida, F.: An effective scheduler for ip routers. In: Proceedings of ISCC'2000, Antibes, France, Fifth IEEE Symposium on Computers and Communications (2000)
[17] Quadros, G., Alves, A., Silva, J., Matos, H., Monteiro, E., Boavida, F.: A queue management system for differentiated-services ip routers. In Crowcroft, J., Roberts, J., Smirnov, M.I., eds.: Lecture Notes in Computer Science. Volume 1922., Berlin, Germany, Springer-Verlag, Heidelberg (2000) 14–27
[18] Alves, A., Quadros, G., Monteiro, E., Boavida, F.: Qostat – a tool for the evaluation of qos-capable routers. In: Proceedings of SPIES's International Symposium on Voice, Video, and Data Communications, Boston (2000)
[19] netIQ: Chariot – User Guide. (2001)
[20] Oliveira, M., Brito, J., Melo, B., Quadros, G., Monteiro, E.: Encaminhamento com qualidade de serviço: Desafios da implementação da estratégia qosr-lct (portuguese). In: Proceedings of CRC'2000, Third National Conference on Computer Networks – Technologies and Applications, Viseu, Portugal, FCCN (2000)
[21] Oliveira, M., Melo, B., Quadros, G., Monteiro, E.: Quality of service routing in the differentiated services framework. In: Proceedings of SPIES's International Symposium on Voice, Video, and Data Communications, Boston (2000)
[22] Lourenco, D., Oliveira, M., Quadros, G., Monteiro, E.: Definição do mecanismo de controlo de admissão para o modelo de serviços de lct-uc. In: Proceedings of CRC'2000, Third National Conference on Computer Networks – Technologies and Applications, Viseu, Portugal, FCCN (2000)
[23] Breslau, L., Shenker, S.: Best–effort versus reservations: A simple comparative analysis. ACM Computer Communication Review **28** (1998) 3–16
[24] Braden, R., Clark, S., Shenker, S.: Integrated services in the internet architecture. RFC 1633, IETF (1994)
[25] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.: An architecture for differentiated services. RFC 2475, IETF (1998)
[26] Shenker, S., Partridge, C., Guerin, R.: Specification of guaranteed quality of service. RFC 2212, IETF (1997)
[27] Wroclawski, J.: Specification of the controlled-load network element service. RFC 2211, IETF (1997)
[28] Braden, R., Zhang, L., Berson, S., Herzog, S., Jamin, S.: Resource ReSerVation protocol (RSVP). RFC 2205, IETF (1997)
[29] Jacobson, V., Nichols, K., Poduri, K.: An expedited forwarding PHB. RFC 2598, IETF (1999)
[30] Heinanen, J., Baker, F., Weiss, W., Wroclawski, J.: Assured forwarding PHB group. RFC 2597, IETF (1999)
[31] Baker, F., Iturralde, C., Le Faucheur, F., Davie, B.: RSVP reservations aggregation. Internet Draft, IETF (2001) (Work in progress).
[32] Bernet, Y.: Format of the RSVP DCLASS object. RFC 2996, IETF (2000)
[33] Bernet, Y., Ford, P., Yavatkar, R., Baker, F., Zhang, L., Speer, M., Braden, R., Davie, B., Wroclawski, J., Felstaine, E.: A framework for integrated services operation over diffserv networks. RFC 2998, IETF (2000)

[34] Cetinkaya, C., Knightly, E.W.: Egress admission control. In: Proc. of IEEE INFOCOM 2000, Tel Aviv, Israel (2000) 1471–1480
[35] Breslau, L., Jamin, S., Shenker, S.: Comments on the performance of measurement–based admission control algorithms. In: Proc. of IEEE INFOCOM 2000, Tel Aviv, Israel (2000) 1233–1242
[36] Karlsson, G.: Providing quality for internet video services. In: Proc. of CNIT/IEEE ITWoDC 98, Ischia, Italy (1998) 133–146
[37] Fodor (née Elek), V., Karlsson, G., Rönngren, R.: Admission control based on end-to-end measurements. In: Proc. of IEEE INFOCOM 2000, Tel Aviv, Israel (2000) 623–630
[38] Ramakrishnan, K., Floyd, S.: A proposal to add explicit congestion notification (ECN) to IP. RFC 2481, IETF (1999)
[39] Kelly, F.P., Key, P.B., Zachary, S.: Distributed admission control. IEEE Journal on Selected Areas in Communications **18** (2000) 2617–2628
[40] Kelly, T.: An ECN probe–based conection acceptance control. ACM Computer Communication Review **31** (2001) 14–25
[41] Bianchi, G., Capone, A., Petrioli, C.: Throughput analysis of end–to–end measurement-based admission control in IP. In: Proc. of IEEE INFOCOM 2000, Tel Aviv, Israel (2000) 1461–1470
[42] Bianchi, G., Capone, A., Petrioli, C.: Packet management techniques for measurement based end-to-end admission control in IP networks. Journal of Communications and Networks **2** (2000) 147–156
[43] Bianchi, G., Borgonovo, F., Capone, A., Petrioli, C.: Endpoint admission control with delay variation measurements for QoS in IP networks. ACM Computer Communication Review **32** (2002) 61–69
[44] Breslau, L., Knightly, E.W., Shenker, S., Stoica, I., Zhang, H.: Endpoint admission control: Architectural issues and performance. In: Proc. of ACM SIGCOMM 2000, Stockholm, Sweden (2000) 57–69
[45] Más Ivars, I., Karlsson, G.: PBAC: Probe–based admission control. In: Proc. of QoFIS 2001, Coimbra, Portugal (2001) 97–109
[46] Más, I., Fodor, V., Karlsson, G.: Probe–based admission control for multicast. In: Proc. of IWQoS 02, Miami Beach, Florida (2002) 97–109
[47] Gibbens, R.J., Kelly, F.P.: Distributed connection acceptance control for a connectionless network. In: Proc. of the 16th International Teletraffic Congress, Edinburgh, Scotland (1999) 941–952
[48] Bianchi, G., Borgonovo, F., Capone, A., Fratta, L., Petrioli, C.: PCP: an end-to-end measurement-based call admission control for real-time services over IP networks. In: Proc. of QoS–IP 2001, Rome, Italy (2001) 391–406
[49] Roberts, J.W., Mocci, U., Virtamo, J., eds.: COST 242: Broadband Network Teletraffic. Volume 1155 of Lecture notes in computer science. Springer–Verlag (1996)
[50] Conte, M., Más, I., Fodor, V., Karlsson, G.: Policy enforcing for probe–based admission control. In: Proc. of NTS 16, Espoo, Finland (2002) 45–55
[51] Ventre, G., et al.: Quality of Service control in Premium IP networks. Deliverable 2.1, IST Project CADENUS — IST 11017 (2001)
[52] Smirnov, M., et al.: SLA Networks in Premium IP. Deliverable 1.1, IST Project CADENUS — IST 11017 (2001)
[53] Goderis, D., et al.: Service Level Specification Semantics and Parameters. Internet Draft, IETF (2002) (Work in progress).
[54] Quittek, J., Zseby, T., Claise, B.: Requirements for IP Flow Information Export. Internet Draft, IETF (2002) (Work in progress).
[55] ebXML Technical Architecture Project Team: ebXML Technical Architecture Specification v.1.0.4. Technical specification, ebXML Consortium (2001)

[56] D'Antonio, S., Fadini, B., Romano, S., Ventre, G.: Designing Service Negotiation Entities for the Electronic Marketplace. In: Proceedings of SEKE2002, Ischia, Napoli — Italy (2002)
[57] Cremonese, P., et al.: A Framework for Policy-based Management of QoS-aware IP Networks. In: Proceedings of Networking2002, Pisa — Italy (2002)
[58] Chan, K., et al.: COPS Usage for Policy Provisioning (COPS-PR). RFC 3084, IETF (2001)
[59] Rawlins, D., et al.: Framework of COPS-PR Policy Usage Feedback. Internet Draft, IETF (2002) (Work in progress).
[60] Zhang, Z., Towsley, D., Kurose, J.: Statistical analysis of the generalized processor sharing scheduling discipline. In Proceedings of ACM SIGCOMM (1994) 68–77
[61] Zhang, Z., Liu, Z., Kurose, J., Towsley, D.: Call admission control schemes under generalized processor sharing scheduling. The Journal of Telecommunication Systems, Modeling, Analysis, Design, and Management **7** (1997)
[62] Elwalid, A., Mitra, D.: Design of generalized processor sharing schedulers which statistically multiplex heterogeneous qos classes. In Proceedings of IEEE INFOCOM '99 (1999) 1220–1230
[63] Parekh, A., Gallager, R.G.: A generalized processor sharing approach to flow control in integrated services networks: The single-node case. IEEE/ACM Transactions on Networking **1** (1993) 344–357
[64] Parekh, A., Gallager, R.G.: A generalized processor sharing approach to flow control in integrated services networks: The multiple node case. IEEE/ACM Transactions on Networking **2** (1994) 137–150
[65] Szabo, R., Barta, P., Nemeth, F., Biro, J.: Call admission control in generalized processor sharing (gps) schedulers using non-rate proportional weighting of sessions. In Proceedings of INFOCOM '00 (2000)
[66] Zhang, Z., Liu, Z., Towsley, D.: Closed-form deterministic end-to-end performance bounds for the generalized processor sharing scheduling discipline. Journal of Combinatorial Optimization **1** (1998)
[67] Georgiadis, L., Gu'erin, R., Peris, V., Sivarajan, K.: Efficient network qos provisioning based on per node traffic shaping. IEEE/ACM Transactions on Networking **4** (1996) 482–501
[68] Duffield, N.G., Lakshman, T.V., Stiliadis, D.: On adaptive bandwidth sharing with rate guarantees. In Proceedings of INFOCOM '98 (1998) 1122–1130
[69] Chang, C.S., Chen, K.C.: Service curve proportional sharing algorithm for service-guaranteed multiaccess in integrated-service distributed networks. In Proceedings of GLOBECOM '99 (1999) 1340 –1344
[70] Stamoulis, A., Giannakis, G.: Deterministic time-varying packet fair queueing for integrated services networks. In Proceedings of GLOBECOM '00 (2000) 621–625
[71] Toutain, F.: Decoupled generalized processor sharing: A fair queueing principle for adaptive multimedia applications. In Proceedings of INFOCOM '98 (1998) 291–298
[72] Panagakis, A., Stavrakakis, I.: Optimal call admission control under generalized processor sharing scheduling. In Proceedings of IWQoS '01 (2001)
[73] Boudec, J.Y.L.: Application of network calculus to guaranteed service networks. IEEE Transactions on Information Theory **44** (1998) 1087–1096
[74] Panagakis, A., Stavrakakis, I.: Generalized processor sharing enhancement through session decomposition. In Proceedings of Net-Con '02 (2002)
[75] Sisodia, G., Headley, M.: Statistical analysis and simulation study of vbr coded video source models in atm networks. In: Proceedings of UPC. (1998) 177–181
[76] Li, S.Q., Hwang, C.L.: Queue response to input correlation functions: discrete spectral analysis. IEEE Trans. on Networking **1** (1997) 533–552

[77] Lombardo, A., Morabito, G., Schembra, G.: An accurate and treatable markov model of mpeg video traffic. In: Proc. of IEEE INFOCOM 1998. (1998) 217–224
[78] MPEG traces archive: (http://www-info3.informatik.uni-wuerzburg.de/mpeg/traces/)
[79] Heyman, D.: The gbar source model for vbr videoconferences. IEEE Trans. on Networking **5** (1997) 554–560
[80] Heyman, D., Tabatabai, A., Lakshman, T.: Statistical analysis and simulation study of video teleconference traffic in atm networks. IEEE Trans. on Circ. and Syst. for Video Tech. **2** (1992) 49–59
[81] Koucheryavy, Y., Moltchanov, D., Harju, J.: A novel two-step mpeg traffic modeling algorithm based on a gbar process. In: Proc. of NET-CON. (2002) 293–304
[82] Lombardo, A., Morabito, G., Palazzo, S., Schembra, G.: Intra-gop modeling of mpeg video traffic. In: Proc. of IEEE ICC. Volume 1. (1998) 563–567
[83] Lombardo, A., Morabito, G., Palazzo, S., Schembra, G.: A fast simulation of mpeg video traffic. In: Proc. GLOBECOM. Volume 2. (1998) 702–707
[84] Blondia, C., Casals, O.: Performance analysis of statistical multiplexing of vbr sources. In: Proc. IEEE INFOCOM. (1992) 828–838
[85] Koucheryavy, Y., Moltchanov, D., Harju, J.: A top-down approach to vod traffc transmission over diffserv domain using the af phb class. In: Proc. of IEEE ICC, Alaska, USA (2003)
[86] Meyer, C.: Matrix analysis and applied linear algebra. SIAM Publications (2000)
[87] Hajek, B., Linhai, H.: On variations of queue response for inputs with the same mean and autocorrelation function. IEEE Trans. on Networking **6** (1998) 588–598
[88] Awduche, D.: MPLS and Traffic Engineering in IP networks. IEEE Communications Magazine **37** (1999) 42–47
[89] Rosen, E., Viswanathan, A., Callon, R.: Multiprotocol label switching architecture. RFC 3031, IETF (2001)
[90] Awduche, D., Chiu, A., Elwalid, A., Widjaja, I., Xiao, X.: Overview and Principles of Internet Traffic Engineering. RFC 3272, IETF (2002)
[91] Blake et al., S.: An architecture for differentiated services. RFC 2475, IETF (1998)
[92] Le Faucheur et al., F.: Multi-protocol label switching (MPLS) support of differentiated services. RFC 3270, IETF (2002)
[93] Cho, K.: Alternate Queuing (ALTQ) module, (http://www.csl.sony.co.jp/person/kjc/programs.html)
[94] Almesberger, W.: Linux network traffic control - implementation overview. White paper, EPFL ICA (2001)
[95] Almesberger, W., Hadi Salim, J., Kuznetsov, A.: Differentiated services on linux. Internet Draft, IETF (1999) (Work in progress).
[96] Leu, J.R.: MPLS for Linux, (http://sourceforge.net/projects/mpls-linux)
[97] Avallone, S., Esposito, M., Pescapé, A., Romano, S., Ventre, G.: Measuring MPLS overhead. In: Proc. of ICCC2002, Mumbai, India (2002)
[98] Avallone, S., D'Arienzo, M., Esposito, M., Pescapé, A., Romano, S., Ventre, G.: Mtools. IEEE Network **16** (2002) 3 Networking column.
[99] Avallone, S., Esposito, M., Pescapé, A., Romano, S., Ventre, G.: Mtools: a one-way delay and round-trip time meter. In Mastorakis, N., Mladenov, V., eds.: Recent Advances in Computers, Computing and Communications. (2002)
[100] Avallone, S., Esposito, M., Pescapé, A., Romano, S., Ventre, G.: An experimental analysis of Diffserv-MPLS interoperability. In: Proc. of ICT 2003, Papeete, French Polynesia (2003)
[101] ITU-T Recommendation G.114: General Characteristics of International Telephone Connections and International Telephone Circuits: One-Way Transmission Time (1998)

[102] Cole, R., Rosenbluth, J.: Voice over IP Performance Monitoring. ACM Computer Communication Review (2002)
[103] ITU-T Recommendation G.107: The E-Model, a computational model for use in transmission planning (1998)
[104] L.F.Sun, G.Wade, B., E.C.Ifeachor: Impact of Packet Loss Location on Perceived Speech Quality. In: Proceedings of 2nd IP-Telephony Workshop (IPTEL '01), Columbia University, New York (2001) 114–122
[105] Kitawaki, N., Kurita, T., Itoh, K.: Effects of Delay on Speech Quality. NTT Review **3** (1991) 88–94
[106] Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.: RTP: A Transport Protocol for Real-Time Applications. RFC 1889, Internet Engineering Task Force (1996) http://www.rfc-editor.org/rfc/rfc1889.txt.
[107] Li, F.: Measurements of Voice over IP Quality. Master's thesis, KTH, Royal Institute of Technology, Sweden (2002)
[108] Hagsand, O., Hansson, K., Marsh, I.: Measuring Internet Telephone Quality: Where are we today ? In: Proceedings of the IEEE Conference on Global Communications (GLOBECOM), Rio, Brazil, IEEE (1999)
[109] Bolot, J., Crepin, H., Garcia, A.: Analysis of audio packet loss in the internet. In: Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV). Lecture Notes in Computer Science, Durham, New Hampshire, Springer (1995) 163–174
[110] Maxemchuk, N.F., Lo, S.: Measurement and interpretation of voice traffic on the Internet. In: Conference Record of the International Conference on Communications (ICC), Montreal, Canada (1997)
[111] Lin, D.: Real-time voice transmissions over the Internet. Master's thesis, Univ. of Illinois at Urbana-Champaign (1999)

Quality of Service Routing

P. Van Mieghem (Ed.)[1], F.A. Kuipers[1], T. Korkmaz[2], M. Krunz[3], M. Curado[4]
E. Monteiro[4], X. Masip-Bruin[5], J. Solé-Pareta[5], and S. Sánchez-López[5]

[1] Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
P.O. Box 5031, 2600 GA Delft, The Netherlands
{P.VanMieghem, F.A.Kuipers}@ewi.tudelft.nl
[2] Department of Computer Science
The University of Texas at San Antonio
6900 North Loop 1604 West, San Antonio, TX 78249, USA
korkmaz@cs.utsa.edu
[3] Department of Electrical & Computer Engineering
University of Arizona
Tucson AZ 85721, USA
Krunz@ece.arizona.edu
[4] Laboratory of Communications and Telematics, CISUC/DEI
University of Coimbra
Polo II, Pinhal de Marrocos, 3030-290 Coimbra, Portugal
{marilia, edmundo}@dei.uc.pt
[5] Department of Computer Architecture
Technical University of Catalonia
Jordi Girona, 1-3, 08034 Barcelona, Catalunya, Spain
{xmasip, pareta, sergio}@ac.upc.es

Abstract. Constraint-based routing is an invaluable part of a full-fledged Quality of Service architecture. Unfortunately, QoS routing with multiple additive constraints is known to be a NP-complete problem. Hence, accurate constraint-based routing algorithms with a fast running time are scarce, perhaps even non-existent. The need for such algorithms has resulted in the proposal of numerous heuristics and a few exact solutions.
This chapter presents a thorough, concise, and fair evaluation of the most important multi-constrained path selection algorithms known today. A performance evaluation of these algorithms is presented based on a complexity analysis and simulation results. Besides the routing algorithm, dynamic aspects of QoS routing are discussed: how to cope with incomplete or inaccurate topology information and (in)stability issues.

1 Introduction

The continuous demand for using multimedia applications over the Internet has triggered a spur of research on how to satisfy the quality of service (QoS) requirements of these applications, e.g., requirements regarding bandwidth, delay,

M. Smirnov et al. (Eds.): COST 263 Final Report, LNCS 2856, pp. 80–117, 2003.

jitter, packet loss, and reliability. These efforts resulted in the proposals of several QoS-based frameworks, such as Integrated Services (Intserv) [11], Differentiated Services (Diffserv) [10], and Multi-Protocol Label Switching (MPLS) [79]. One of the key issues in providing QoS guarantees is *how to determine paths that satisfy QoS constraints*. Solving this problem is known as QoS routing or constraint-based routing.

The research community has extensively studied the QoS routing problem, resulting in many QoS routing algorithms. In this chapter, we provide an overview and performance evaluation for unicast[6] QoS routing algorithms, which try to find a path between a *source* node and a *destination* node that satisfies a set of constraints.

Routing in general involves two entities, namely the *routing protocol* and the *routing algorithm*. The *routing protocol* manages the dynamics of the routing process: capturing the state of the network and its available network resources and distributing this information throughout the network. The *routing algorithm* uses this information to compute paths that optimize a criterion and/or obey constraints. Current best-effort routing consists of shortest path routing that optimizes the sum over the constituent links of a single measure like hopcount or delay. QoS routing takes into account multiple QoS requirements, link dynamics, as well as the implication of the selected routes on network utilization, turning QoS routing into a notoriously challenging problem. Despite its difficulty, we argue that QoS routing is invaluable in a network architecture that needs to satisfy traffic and service requirements. For example, in the context of ATM (PNNI), QoS routing is performed by source nodes to determine suitable paths for connection requests. These connection requests specify QoS constraints that the path must obey. Since ATM is a connection-oriented technology, a path selected by PNNI will remain in use for a potentially long period of time. It is therefore important to choose a path with care. The Intserv/RSVP framework is also able to guarantee some specific QoS constraints. However, this framework relies on the underlying IP routing table to reserve its resources. As long as this routing table is not QoS-aware, paths may be assigned that cannot guarantee the constraints, which will result in blocking. In MPLS, which is a convergence of several efforts aimed at combining the best features of IP and ATM, a source node selects a path, possibly subject to QoS constraints, and uses a signaling protocol (e.g. RSVP or CR-LDP) to reserve resources along that path. In the case of Diffserv, QoS-based routes can be requested, for example, by network administrators for traffic engineering purposes. Such routes can be used to ensure a certain service level agreement [98]. These examples all indicate the importance of constraint-based routing algorithms, both in ATM and IP. Depending on the frequency at which constrained paths are requested, the computational complexity of finding a path subject to multiple constraints is often a complicating but decisive factor.

[6] Multicast QoS routing faces different conceptual problems as discussed in [51]. An overview of several multicast QoS algorithms was given in [81] and more recently in [93].

To enable QoS routing, it is necessary to implement state-dependent, QoS-aware networking protocols. Examples of such protocols are PNNI [7] of the ATM Forum and the QoS-enhanced OSPF protocol [5]. For the first task in routing (i.e., the representation and dissemination of network-state information), both OSPF and PNNI use link-state routing, in which every node tries to acquire a "map" of the underlying network topology and its available resources via flooding. Despite its simplicity and reliability, flooding involves unnecessary communications and causes inefficient use of resources, particularly in the context of QoS routing that requires frequent distribution of multiple, dynamic parameters, e.g., using triggered updates [3]. Designing efficient QoS routing protocols is still an open issue that needs to be investigated further. Hereafter in Sections 2 and 3, we assume that the network-state information is temporarily static and has been distributed throughout the network and is accurately maintained at each node using QoS link-state routing protocols. Once a node possesses the network-state information, it performs the second task in QoS routing, namely computing paths based on multiple QoS constraints. In this chapter, we focus on this so-called *multi-constrained path* selection problem and consider numerous proposed algorithms. Before giving the formal definition of the *multi-constrained path* problem, we explain the notation that is used throughout this chapter.

Let $G(N, E)$ denote a network topology, where N is the set of nodes and E is the set of links. With a slight abuse of notation, we also use N and E to denote the number of nodes and the number of links, respectively. The number of QoS measures (e.g., delay, hopcount, etc.) is denoted by m. Each link is characterized by an m-dimensional link weight vector, consisting of m non-negative QoS weights ($w_i(u, v)$, $i = 1, ..., m$, $(u, v) \in E$) as components. A QoS measure of a path can either be additive (e.g., delay, jitter, the logarithm of 1 minus the probability of packet loss), in which case the weight of that measure equals the sum of the QoS weights of the links defining that path. Or the weight of a QoS measure of a path can be the minimum(maximum) of the QoS weights along the path (e.g., available bandwidth and policy flags). Constraints on min(max) QoS measures can easily be treated by omitting all links (and possibly disconnected nodes) which do not satisfy the requested min(max) QoS constraints. We call this topology filtering. In contrast, constraints on additive QoS measures cause more difficulties. Hence, without loss of generality, we assume all QoS measures to be additive.

The basic problem considered in this chapter can be defined as follows:

Definition 1 *Multi-Constrained Path (MCP) problem:* Consider a network $G(N, E)$. Each link $(u, v) \in E$ is specified by a link-weight vector of m additive QoS weights $w_i(u, v) \geq 0$, $i = 1, ..., m$. Given m constraints L_i, $i = 1, ..., m$, the problem is to find a path P from a source node s to a destination node d such that $w_i(P) \stackrel{def}{=} \sum_{(u,v)\in P} w_i(u, v) \leq L_i$ for $i = 1, ..., m$.

A path that satisfies all m constraints is often referred to as a feasible path. There may be multiple different paths in the graph $G(N, E)$ that satisfy the constraints. According to **Definition 1**, any of these paths is a solution to the

MCP problem. However, it might be desirable to retrieve the path with smallest length $l(P)$ from the set of feasible paths. This problem is called the *multi-constrained optimal path* problem and is formally defined as follows:

Definition 2 *Multi-Constrained Optimal Path (MCOP) problem:* Consider a network $G(N, E)$. Each link $(u, v) \in E$ is specified by a link-weight vector of m additive QoS weights $w_i(u, v) \geq 0$, $i = 1, ..., m$. Given m constraints L_i, $i = 1, ..., m$, the problem is to find a path P from a source node s to a destination node d such that:
(i) $w_i(P) \stackrel{def}{=} \sum_{(u,v)\in P} w_i(u, v) \leq L_i$ for $i = 1, ..., m$
(ii) $l(P) \leq l(P^*)$, $\forall P^*, P$ satisfying (i)

where $l(P)$ can be any function of the weights $w_i(P)$, $i = 1, ..., m$, provided it obeys the criteria for "length" or "distance" in vector algebra (see [87], Appendix A). Minimizing a properly chosen length function can result in an efficient use of the network resources and/or result in a reduction of monetary cost.

In general, MCP, irrespective of path optimization, is known to be an NP-complete problem [26]. Because MCP and MCOP are NP-complete, they are considered to be intractable for large networks. Accordingly, mainly heuristics have been proposed for these problems. In Section 2, the lion's share of the published QoS algorithms is briefly described and compared based on extensive simulations. Complexity will be an important criterion for evaluating the algorithms. Complexity refers to the intrinsic minimum amount of resources needed to solve a problem or execute an algorithm. Complexity consists of *time* complexity and *space* complexity. Only the worst-case *computational time-complexity* and the *average execution time* are considered here. There can be a significant difference between these complexities. Kuipers and Van Mieghem [50] demonstrated that under certain conditions and on average, the MCP problem can be solved in polynomial time despite its worst-case NP-complete complexity. Moreover, there exist specific classes of graphs for which the MCP problem is not NP-complete at all, e.g. if each node has only two neighbors [52].

This chapter follows the two-part structure of routing: the first two sections concentrate on the routing algorithm, while the remaining sections emphasize the routing dynamics. In Section 2 we present an overview of the most important MCP algorithms. Section 3 continues with a performance evaluation of the algorithms listed in Section 2, and based on the simulation results deduces the fundamental concepts involved in QoS routing. The origins of incomplete or inaccurate topology state information are explained in Section 4. Section 5 provides an overview for QoS protocols and Section 6 treats stability of QoS routing. Finally, Section 7 concludes the chapter and lists open issues.

2 Overview of MC(O)P Algorithms

As a prerequisite to the subsequent discussion, we assume the reader is familiar with single-parameter shortest path algorithms (e.g., Dijkstra's and the Bellman-Ford algorithms). Cormen *et al.* [18] provided an excellent introduction on such

algorithms. On the more theoretical level, the monumental treatise of Schrijver [83] encompasses nearly the whole field of combinatorial optimization. Of particular interest to QoS routing algorithms is the in-depth treatment of NP-completeness, polynomial time algorithms, and path and flow algorithms.

2.1 Jaffe's Approximate Algorithm

Jaffe [36] presented two MCP algorithms. The first is an exact pseudo-polynomial-time algorithm with a worst-case complexity of $O(N^5 b \log Nb)$, where b is the largest weight in the graph. Because of this prohibitive complexity, only the second algorithm, hereafter referred to as Jaffe's algorithm, is discussed. Jaffe proposed using a shortest path algorithm on a linear combination of the two link weights:

$$w(u,v) = d_1 \cdot w_1(u,v) + d_2 \cdot w_2(u,v) \tag{1}$$

where d_1 and d_2 are positive multipliers.

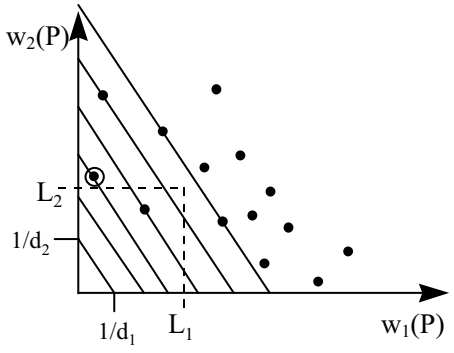


Fig. 1. Representation of the link weight vector $\boldsymbol{w}(P)$ in two dimensions. The dotted rectangle spanned by the constraints L_1 and L_2 forms the space of feasible paths. Jaffe's scanning procedure first encounters the encircled node, which is the path with minimal length.

Each line in Figure 1 shows equilength paths with respect to (w.r.t.) the linear length definition (1). Jaffe's algorithm searches the path weight space along parallel lines specified by $w(P) = c$. As soon as this line hits a path represented by the encircled black dot, the algorithm returns this path as the shortest w.r.t. the linear length definition (1). Figure 1 also illustrates that the shortest path based on a linear combination of link weights does not necessarily reside within the constraints. Jaffe had also observed this fact, so he subsequently provided a nonlinear path length function of the form $f(P) = \max\{w_1(P), L_1\} + \max\{w_2(P), L_2\}$, whose minimization can guarantee finding a feasible path if such a path exists. However, because no simple shortest path algorithm can cope

with this nonlinear length function, Jaffe approximated the nonlinear length by the linear length function (1). Andrew and Kusuma [1] generalized Jaffe's analysis to an arbitrary number of constraints, m, by extending the linear length function to

$$l(P) = \sum_{i=1}^{m} d_i w_i(P) \tag{2}$$

and the nonlinear function to

$$f(P) = \sum_{i=1}^{m} \max\left(w_i(P),\, L_i\right)$$

For the simulations in Section 3, we have used $d_i = \frac{1}{L_i}$ which maximizes the volume of the solution space that can be scanned by linear equilength lines (2) subject to $w_i(P) \leq L_i$. Furthermore, we have used Dijkstra's algorithm [22] with Fibonacci heaps, leading to a complexity for Jaffe's algorithm of $O(N \log N + mE)$.

If the returned path is not feasible, then Jaffe's algorithm stops, although the search could be continued by using different values for d_i, which might result in a feasible path. Unfortunately, in some cases, even if all possible combinations of d_i are exhausted, a feasible path may not be found using linear search. As shown in [87], an exact algorithm must necessarily use a nonlinear length function, even though a nonlinear function cannot be minimized with a simple shortest path algorithm.

2.2 Iwata's Algorithm

Iwata *et al.* [35] proposed a polynomial-time algorithm to solve the MCP problem. The algorithm first computes one (or more) shortest path(s) based on one QoS measure and then checks if all the constraints are met. If this is not the case, the procedure is repeated with another measure until a feasible path is found or all QoS measures are examined. A similar approach has been proposed by Lee *et al.* [55]. In the simulations we only evaluate Iwata's algorithm.

The problem with Iwata's algorithm is that there is no guarantee that any of the shortest paths w.r.t. each individual measure is close to a path within the constraints. This is illustrated in Figure 2, which shows the twenty shortest paths of a two-constraint problem applied to a random graph with 100 nodes. Only the second and third shortest path for measure 1 and the second and fourth shortest path for measure 2 lie within the constraints.

In our simulations we will only consider one shortest path per QoS measure computed via Dijkstra's algorithm, leading to a complexity of $O(mN \log N + mE)$.

2.3 SAMCRA: A Self-adaptive Multiple Constraints Routing Algorithm

SAMCRA [87] is a successor of TAMCRA, a Tunable Accuracy Multiple Constraints Routing Algorithm [21,20]. TAMCRA and SAMCRA are based on three

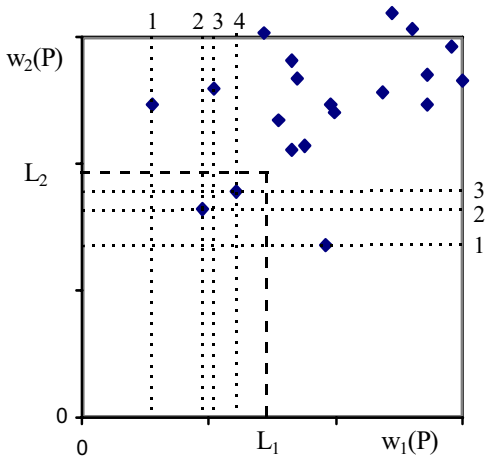


Fig. 2. Twenty shortest paths for a two-constraint problem. Each path is represented as a dot and the coordinates of each dot are its path-length for each measure individually.

fundamental concepts: (1) a nonlinear measure for the path length, (2) a k-shortest path approach [17], and (3) the principle of non-dominated paths [33]. These three principles are explained next.

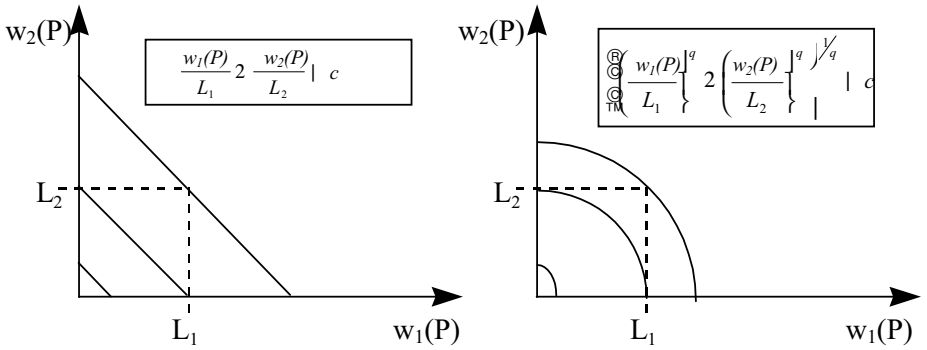


Fig. 3. Scanning procedure with (a) straight equilength lines. (b) curved equilength lines..

1. *Nonlinear path-length measure.* Figure 3 illustrates that the curved equilength lines of a nonlinear length function scan the constraints area in a more efficient way than the linear equilength lines of linear length definitions. The formula in Figure 3b is derived from Holder's q-vector norm [27].

Ideally, the equilength lines should perfectly match the boundaries of the constraints. Scanning the constraint area without ever selecting a solution outside the constraint area is only achieved when $q \to \infty$. Motivated by the geometry of the constraints surface in m-dimensional space, the length of a path P is defined, equivalent to Holder's q-vector norm with $q \to \infty$, as follows [21]:

$$l(P) = \max_{1 \le i \le m} \left(\frac{w_i(P)}{L_i} \right) \tag{3}$$

A solution to the MCP problem is a path whose weights are all within the constraints: $l(P) \le 1$. Depending on the specifics of a constrained optimization problem, SAMCRA can be used with different length functions, provided they obey the criteria for length in vector algebra. Examples of length functions were given in [87]. The length function (3) treats all QoS measures as equally important. An important corollary of a nonlinear path length such as (3) is that *the subsections of shortest paths in multiple dimensions are not necessarily shortest paths*. This suggests considering more than the shortest path, leading us to the k-shortest path approach.

2. *k-shortest path algorithm.* This algorithm (e.g., as presented in [17]) is essentially Dijkstra's algorithm, with extensions to return not only the shortest path to a given destination but also the second shortest, the third shortest, ..., the kth shortest path. In SAMCRA the k-shortest path concept is applied to intermediate nodes i on the path from the source node s to the destination node d to keep track of multiple sub-paths from s to i. Not all sub-paths are stored, but the search space is reduced by applying the principle of non-dominance.
3. *Principle of non-dominance.* A path Q is said to be dominated by a path P if $w_i(P) \le w_i(Q)$ for $i = 1, .., m$, with an inequality for at least one i. SAMCRA only considers non-dominated (sub)-paths. This property allows it to efficiently reduce the search space without compromising the solution. "Dominance" can be regarded as a multidimensional relaxation. The latter is a key aspect of single-parameter shortest path algorithms (such as Dijkstra and Bellman-Ford).

Both SAMCRA and TAMCRA have a worst-case complexity of

$$O(kN \log(kN) + k^2 mE)$$

For TAMCRA the number of paths considered during execution (k) is fixed, and hence the complexity is polynomial. SAMCRA self-adaptively controls this k, which can grow exponentially in the worst case. Knowledge about k is crucial to the complexity of SAMCRA. An upper bound on k is $k_{\max} = \lfloor e(N-2)! \rfloor$, which is a bound on the total number of paths between a source and a destination in $G(N, E)$ [88]. If the constraints/measures have a finite granularity, another upper bound applies:

$$k_{\max} = \min\left(\frac{\prod_{i=1}^{m} L_i}{\max_j(L_j)}, \lfloor e(N-2)! \rfloor\right)$$

where the constraints L_i are expressed as an integer number of a basic unit.

The self-adaptivity in k makes SAMCRA an exact MCOP algorithm: SAMCRA guarantees finding the shortest path within the constraints provided that such a path exists. In this process, SAMCRA only allocates queue space when truly needed and self-adaptively adjusts the number of stored paths k in each node. In TAMCRA the allocated queue space is predefined via k. In our simulations, we choose $k = 2$ for TAMCRA since this small value already produces good results. Of course, better performance can be achieved with a larger k. Simulation results for different values for k can be found in [21].

Kuipers and Van Mieghem [53] have proposed an exact hybrid MCP algorithm, that integrates the speed of TAMCRA with the exactness of SAMCRA. This hybrid algorithm called HAMCRA may be considered a form of bi-directional search. Bi-directional search is a powerful method in one-dimensional unicast routing, but is unfortunately not fully extendible to m-dimensional routing [53]. Finally, a link-disjoint paths algorithm DIMCRA [31], a heuristic derived from SAMCRA, returns two link-disjoint paths both obeying a same constraints vector.

2.4 Chen's Approximate Algorithm

Chen and Nahrstedt [12] provided an approximate algorithm for the MCP problem. This algorithm first transforms the MCP problem into a simpler problem by scaling down $m-1$ (real) link weights to integer weights as follows:

$$w_i^*(u,v) = \left\lceil \frac{w_i(u,v) \cdot x_i}{L_i} \right\rceil \text{ for } i = 2, 3, \ldots, m,$$

where x_i are predefined positive integers. The simplified problem consists of finding a path P for which $w_1(P) \leq L_1$ and $w_i^*(P) \leq x_i$, $2 \leq i \leq m$. A solution to this simplified problem is also a solution to the original MCP problem, but not vice versa (because the conditions of the simplified problem are more strict). Since the simplified problem can be solved exactly, Chen and Nahrstedt have shown that *the MCP problem can be exactly solved in polynomial time provided that at least $m-1$ QoS measures have bounded integer weights.*

To solve the simplified MCP problem, Chen and Nahrstedt proposed two algorithms based on dynamic programming: the Extended Dijkstra's Shortest Path algorithm (EDSP) and the Extended Bellman-Ford algorithm (EBF). The algorithms return a path that minimizes the first (real) weight provided that the other $m-1$ (integer) weights are within the constraints. The EBF algorithm is expected to give better performance in terms of execution time when the graph is sparse and the number of nodes is relatively large. We have chosen to implement the EBF version for our simulations.

The complexities of EDSP and EBF are $O(x_2^2 \cdots x_m^2 N^2)$ and $O(x_2 \cdots x_m NE)$, respectively. To achieve good performance, large x_i's are needed, which makes

this approach rather computationally intensive for practical purposes. By adopting the concept of non-dominance, like in SAMCRA, this algorithm could[7] reduce its search space, resulting in a faster execution time.

2.5 Randomized Algorithm

Korkmaz and Krunz [48] proposed a randomized heuristic for the MCP problem. The concept behind randomization is to make random decisions during the execution of an algorithm [68] so that unforeseen traps can potentially be avoided when searching for a feasible path. The proposed randomized algorithm is divided into two parts: an initialization phase and a randomized search. In the initialization phase, the algorithm computes the shortest paths from every node u to the destination node d w.r.t. each QoS measure and w.r.t. the linear combination of all m measures. This information will provide lower bounds for the path weight vectors of the paths from u to d. Based on the information obtained in the initialization phase, the algorithm can decide whether there is a chance of finding a feasible path or not. If so, the algorithm starts from the source node s and explores the graph using a randomized breadth-first search (BFS). In contrast to the conventional BFS, which systematically discovers every node that is reachable from node s, the randomized BFS discovers nodes from which there is a good chance to reach the destination d. By using the information obtained in the initialization phase, the randomized BFS can check whether this chance exists before discovering a node. If there is no chance of reaching the destination, the algorithm foresees the trap and avoids exploring such nodes any further. We will refer to this search-space reducing technique as the look-ahead property. The objectives of the look-ahead property are twofold. First, the lower-bound vectors obtained in the initialization phase are used to check whether a sub-path from s to u can become a feasible path. This is a search-space reducing technique. Second, a different preference rule for extracting nodes can be adopted based on the predicted end-to-end length, i.e. the length of the sub-path weight vector plus the lower bound vector. The randomized BFS continues searching by randomly selecting discovered nodes until the destination node is reached. If the randomized BFS fails in the first attempt, it is possible to execute only the randomized BFS again so that the probability of finding a feasible path can be increased.

Under the same network conditions, multiple executions of the randomized algorithm may return different paths between the same source and destination pair, providing some load balancing. However, some applications might require the same path again. In such cases, path caching can be used [76].

The worst-case complexity of the randomized algorithm is $O(mN \log N + mE)$. For our simulations, we only executed one iteration of the randomized BFS.

[7] In Section 3 we have simulated all algorithms in their original forms, without any possible improvements.

2.6 H_MCOP

Korkmaz and Krunz [49] also provided a heuristic called H_MCOP. This heuristic tries to find a path within the constraints by using the nonlinear path length function (3) of SAMCRA. In addition, H_MCOP tries to simultaneously minimize the weight of a single "cost" measure along the path. To achieve both objectives simultaneously, H_MCOP executes two modified versions of Dijkstra's algorithm in the backward and forward directions. In the backward direction, H_MCOP uses Dijkstra's algorithm for computing the shortest paths from every node to the destination node d w.r.t. $w(u,v) = \sum_{i=1}^{m} \frac{w_i(u,v)}{L_i}$. Later on, these paths from every node u to the destination node d are used to estimate how suitable the remaining sub-paths are. In the forward direction, H_MCOP uses a modified version of Dijkstra's algorithm. This version starts from the source node s and discovers each node u based on a path P, where P is a heuristically determined complete s-d path that is obtained by concatenating the already traveled sub-path from s to u and the estimated remaining sub-path from u to d. Since H_MCOP considers complete paths before reaching the destination, it can foresee several infeasible paths during the search. If paths seem feasible, then the algorithm can switch to explore these feasible paths based on the minimization of the single measure. Although similar to the look-ahead property, this technique only provides a preference rule for choosing paths and cannot be used as a search-space reducing technique.

The complexity of the H_MCOP algorithm is $O(N \log N + mE)$. If one deals only with the MCP problem, then H_MCOP should be stopped whenever a feasible path is found during the search in the backward direction, reducing the computational complexity. The performance of H_MCOP in finding feasible paths can be improved by using the k-shortest path algorithm and by eliminating dominated paths.

2.7 Limited Path Heuristic

Yuan [99] presented two heuristics for the MCP problem. The first "limited granularity" heuristic has a complexity of $O(N^m E)$, whereas the second "limited path" heuristic (LPH) has a complexity of $O(k^2 NE)$, where k corresponds to the queue size at each node. The author claims that when $k = O(N^2 \log_2 N)$, the limited path heuristic has a very high probability of finding a feasible path, provided that such a path exists. However, applying this value results in an excessive execution time.

The performance of both algorithms is comparable when $m \leq 3$. For $m > 3$, LPH performs better than the limited granularity heuristic. Hence, we will only evaluate LPH. Another reason for not considering the limited granularity heuristic is that it closely resembles the algorithm of Chen and Nahrstedt (Section 2.4).

LPH is an extended Bellman-Ford algorithm that uses two of the fundamental concepts of TAMCRA. Both use the concept of non-dominance and maintain at most k paths per node. However, TAMCRA uses a *k-shortest* path approach, while LPH stores the first (and not necessarily shortest) k paths. Furthermore

LPH does not check whether a sub-path obeys the constraints, but only at the end for the destination node. An obvious difference is that LPH uses a Bellman-Ford approach, while TAMCRA uses a Dijkstra-like search. Simulations reveal that Bellman-Ford-like implementations require more execution time than Dijkstra-like implementations, especially when the graphs are dense. To conform with the queue size allocated for TAMCRA, we set $k = 2$ in the simulations for LPH.

2.8 A*Prune

Liu and Ramakrishnan [57] considered the problem of finding not only one but multiple (K) shortest paths satisfying the constraints. The length function used is the same as Jaffe's length function (2). The authors proposed an exact algorithm called A*Prune. If there are no K feasible paths present, the algorithm will only return those that are within the constraints. For the simulations we took $K = 1$.

For each QoS measure, A*Prune first calculates the shortest paths from the source s to all nodes $i \in N\backslash\{s\}$ and from the destination d to all nodes $i \in N\backslash\{d\}$. The weights of these paths will be used to evaluate whether a certain sub-path can indeed become a feasible path (similar look-ahead features were also used in [48]). After this initialization phase, the algorithm proceeds in a Dijkstra-like fashion. The node with the shortest predicted end-to-end length[8] is extracted from a heap and then all of its neighbors are examined. The neighbors that cause a loop or lead to a violation of the constraints are pruned. The A*Prune algorithm continues extracting/pruning nodes until K constrained shortest paths from s to d are found or until the heap is empty.

If Q is the number of stored paths, then the worst-case complexity is $O(QN(m + h + \log Q))$, where h is the number of hops of the retrieved path. This complexity is exponential, because Q can grow exponentially with $G(N, E)$. The authors in [57] indicated that it is possible to implement a Bounded A*Prune algorithm with a polynomial-time complexity, at the risk of losing exactness.

2.9 "Specialized" QoS Routing Algorithms

Several works in the literature have aimed at addressing special yet important sub-problems in QoS routing. For example, researchers addressed QoS routing in the context of bandwidth and delay. Routing with these two measures is not NP-complete. Wang and Crowcroft [95] presented a *bandwidth-delay based routing algorithm*, which simply prunes all links that do not satisfy the bandwidth constraint and then finds the shortest path w.r.t. delay in the pruned graph. A much researched problem is the NP-complete *Restricted Shortest Path* (RSP) problem. The RSP problem only considers two measures, namely delay and cost.

[8] The length function is a linear function of all measures (2). If there are multiple sub-paths with equal predicted end-to-end lengths, the one with the so-far shortest length is chosen.

The problem consists of finding a path from s to d for which the delay obeys a given constraint and the cost is minimum. In the literature, the RSP problem is also studied under different names such as the delay-constrained least-cost path, minimum-cost restricted-time path, and constrained shortest path. Many heuristics were proposed for this problem, e.g. [32,78,39,30]. Several path selection algorithms based on different combinations of bandwidth, delay, and hopcount were discussed in [74] (e.g. widest-shortest path and shortest-widest path). In addition, new algorithms were proposed to find more than one feasible path w.r.t. bandwidth and delay (e.g. Maximally Disjoint Shortest and Widest Paths) [86]. Kodialam and Lakshman [44] proposed bandwidth guaranteed dynamic routing algorithms. Orda and Sprintson [75] considered pre-computation of paths with minimum hopcount and bandwidth guarantees. They also provided some approximation algorithms that take into account certain constraints during the pre-computation. Guerin and Orda [29] focused on the impact of advance reservation on the path selection process. They described possible extensions to path selection algorithms in order to make them advance-reservation aware, and evaluated the added complexity introduced by these extensions. Fortz and Thorup [24] investigated how to set link weights based on previous measurements so that the shortest paths can provide better load balancing and can meet the desired QoS constraints. The path selection problem becomes simpler when dependencies exist between the QoS measures, for example as a result of implementing specific scheduling schemes at network routers [60]. Specifically, if Weighted Fair Queueing (WFQ) scheduling is being used and the constraints are on bandwidth, queueing delay, jitter, and loss, then the problem can be reduced to a standard shortest path problem by representing all the constraints in terms of bandwidth. However, although queueing delay can be formulated as a function of bandwidth, this is not the case for the propagation delay, which cannot be ignored in high-speed networks.

3 Performance Analysis of MCP Algorithms

3.1 Comparison of MCP Algorithms

In this section, we present and discuss our simulations results for the MCP problem. The previously presented algorithms were ran several times on different realizations of the Waxman topology [97,88]. Waxman graphs have the attractiveness of being easy to generate, allowing us to evaluate the underlying algorithms on many such graphs. This is crucial in an algorithmic study where it is necessary to evaluate many scenarios in order to be able to draw reliable conclusions. As shown in [88], the conclusions reached for the Waxman graphs are also valid for the class of random graphs $G_p(N)$. All simulations consisted of generating 10^4 topologies. The values of the m link weights were sampled from independent uniform distributions in the range (0,1).

The choice of the constraints is important because it determines how many (if any) feasible paths exist. We adopt two sets of constraints, referred to as $L1$ and $L2$:

- $L1$: First SAMCRA was used with all $L_i = N$ for $1 \leq i \leq m$. From that resulting shortest path P according to (3) the constraints were set as $L_i = w_i(P)$ for $1 \leq i \leq m$.
- $L2$: $L_i = \max_{j=1,...,m}(w_i(SP_j)),\ i = 1, ..., m$, where SP_j is the shortest path based on the j-th measure.

The first set of constraints obtained with the exact algorithm SAMCRA is very strict; there is only one feasible path present in the graph. The second set of constraints ($L2$) is based on the weights of the shortest paths for each QoS measure. We use Dijkstra to compute these shortest paths and for each of these m paths we store their path weight vectors. We then choose for each measure i the maximum ith component of these m path weight vectors. With these constraints, the MCP problem can be shown to be polynomial [52]. (Iwata's algorithm can always find a feasible path with this set of constraints)

From the simulation runs, we obtain the *success rate* and the *normalized execution time*. The *success rate* of an algorithm is defined as the number of feasible paths found divided by the number of examined graphs. The *normalized execution time* of an algorithm is defined as the *average execution time* of the algorithm (over all examined graphs) divided by the *execution time* of Dijkstra's algorithm.

Our simulations revealed that the Bellman-Ford-like algorithms (Chen's algorithm and the Limited Path Heuristic) consume significantly more *execution time* than their Dijkstra-like counterparts. We therefore omitted them from the results presented in this chapter.

Figure 4 gives the *success rate* for four different topology sizes (N = 50, 100, 200 and 400) with $m = 2$. The exact algorithms SAMCRA and A*Prune always give the highest *success rate* possible. The difference in the *success rate* of the heuristics is especially noticeable when the constraints are strict. In this case, Jaffe's and Iwata's algorithms perform significantly worse than the others. The only heuristic that is not affected much by strict constraints is the randomized algorithm. However, its *execution time* is comparable to that of the exact algorithms.

Figure 5 displays the *normalized execution time*. It is interesting to observe that the *execution time* of the exact algorithm SAMCRA does not deviate much from the polynomial time heuristics. This difference increases with the number of nodes, but an exponentially growing difference is not noticeable! A first step towards understanding this phenomenon was provided by Kuipers and Van Mieghem in [50] and [52]. Furthermore, when the constraints are less stringent, the *execution time* increases. This applies to the algorithms that try to minimize some length function in addition to satisfying the constraints (i.e., MCOP algorithms). When the constraints are loose, there are more feasible paths in the network, from which the shortest one is to be found. Searching through this larger set results in an increased *execution time*. If optimization is not strived for (MCP), then it is easier to find a feasible path under loose than under strict constraints.

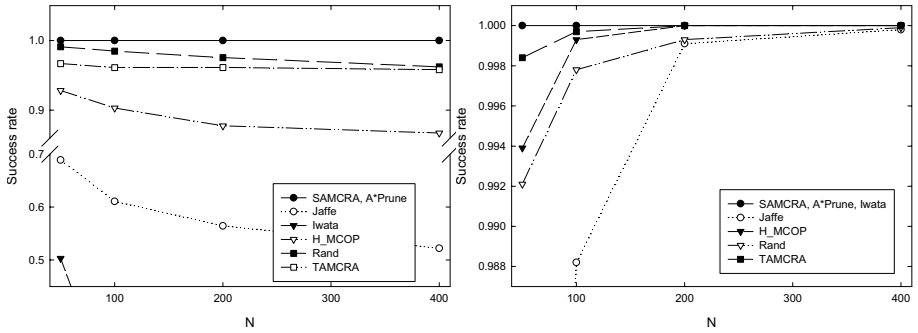


Fig. 4. Success rate versus number of nodes for $m = 2$. The results for the set of constraints $L1$ is depicted on the left and for $L2$ on the right.

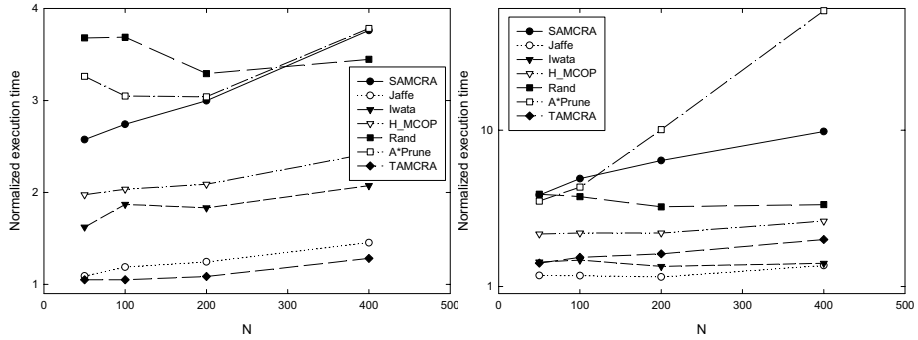


Fig. 5. Normalized execution time versus N for $m = 2$. The results for the set of constraints $L1$ are plotted on the left and for $L2$ on the right.

We have also simulated the performance of the algorithms as a function of m ($m = 2$, 4, 8 and 16). The results are plotted in Figures 6 and 7. The algorithms display a similar ranking in the *success rate* as in Figure 4. All link weights are independent uniformly distributed random variables. When the link weights are independent, a larger m implies a larger number of non-dominated paths to evaluate. However, at a certain value of $m*$, the constraint values will dominate, leading to an increasing number of paths that violate the constraints. Hence less paths need to be evaluated. This property is explained in [87], where the following theorem was proved: If the m components of the link weight vector are independent random variables and the constraints L_j are such that $0 \leq w_j/L_j \leq 1$, any path with K hops has precisely a length (as defined in (3)) equal to K in the limit $m \to \infty$. This theorem means that, for $m \to \infty$ and independent link weight components, the m-dimensional problem reduces to a single metric problem where the path that minimizes the hopcount is also the

shortest path. The impact of the constraint values also follows by comparing the execution times in Figures 6 and 7. If the constraints are loose, then there is a significant difference in *execution time* between the exact algorithms SAMCRA and A*Prune. This can be attributed to the look-ahead property of A*Prune, which can foresee whether sub-paths can lead to feasible end-to-end paths. Again, note that the *execution times* do not exhibit any NP-complete behavior.

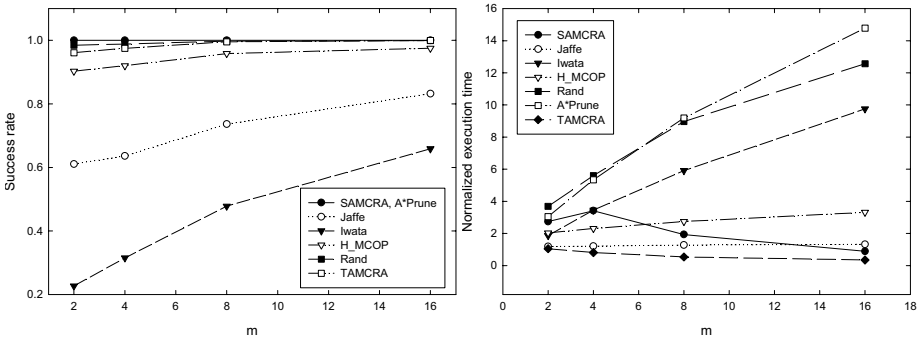


Fig. 6. Success rate and normalized execution time versus m in a 100-node network with the set of constraints $L1$.

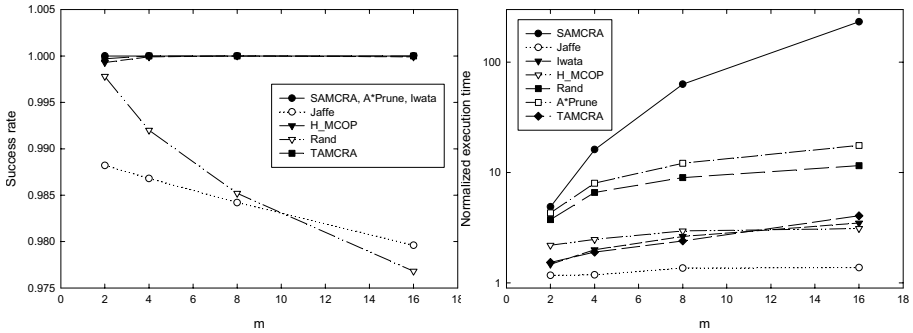


Fig. 7. Success rate and normalized execution time versus m in a 100-node network with the set of constraints $L2$.

Based on these results we can rank the heuristics according to their *success rate* and *execution time* as follows: TAMCRA, H_MCOP, Randomized algorithm, Jaffe's algorithm, and finally Iwata's algorithm. The simulation results presented in [49] displayed a higher *success rate* for H_MCOP than for TAMCRA. This was due to a programming error, where the forward search of H_MCOP was

revisiting the previously explored nodes (which is similar to using $k > 1$ in the k-shortest-paths-based algorithms). This implementation bug has now been fixed, which resulted in a better *success rate* for TAMCRA.

3.2 Summary of the Performance of MCP Algorithms

Based on the simulation results of the previous section, the strengths of these algorithms are summarized. The conclusions are only valid for the considered class of graphs, namely the Waxman graphs (also random graphs [88]) with independent uniformly distributed link weights, but might also hold for other classes of graphs.

For the MCP problem, we observe that TAMCRA-like algorithms have a higher *success rate* than linear approximations and Bellman-Ford based algorithms. This higher *success rate* is attributed to the following concepts:

1. *Using a Dijkstra-like search along with a nonlinear length function.*
   A nonlinear length function is a prerequisite for exactness. When the link weights are positively correlated, a linear approach may give a high *success rate* in finding feasible paths, but under different circumstances the returned path may violate the constraints by 100%.
   A Bellman-Ford-like search usually runs faster on sparse graphs than on dense ones. However, our simulations indicate that even on sparse graphs, a Dijkstra-like heap-optimized search runs significantly faster.
2. *Tunable accuracy through a k-shortest path functionality.*
   Routing with multiple constraints may require that multiple paths be stored at a node, necessitating a k-shortest path approach.
3. *Reducing the search space through the concept of non-dominance.*
   Reducing the search space is always desirable as it reduces the *execution time* of an algorithm. The non-dominance principle is a strong search-space reducing technique, especially when the number of constraints is small. Note that the constraints themselves, if strict, also provide a search-space reduction, since many sub-paths will violate those constraints.
4. *Predicting the feasibility of paths (look-ahead property).*
   First calculating a path in polynomial time between the source and destination and then using this information to find a feasible path between the same source and destination is especially useful when graphs become "hard to solve", i.e. N, E and m are large. This look-ahead property allows us to compute lower bounds on end-to-end paths, which can be used to check the feasibility of paths. Moreover, better preference rules can be adopted to extract nodes from the queue.

The exactness of TAMCRA-like algorithms depends on the flexibility in choosing k. If k is not restricted, then both MCP and MCOP problems can be solved exactly, as done by SAMCRA. Although k is not restricted in SAMCRA, simulations on Waxman graphs with independent uniformly distributed random link weights show that the *execution time* of this exact algorithm increases linearly with the number of nodes, providing a scalable solution to the MC(O)P

problem. If a slightly larger *execution time* is permitted, then such exact algorithms are a good option. Furthermore, simulations show that TAMCRA-like algorithms with small values of k render near-exact solutions with a Dijkstra-like complexity. For example, TAMCRA with $k = 2$ has almost the same *success rate* as the exact algorithms.

4 Influence of Network Dynamics on QoS Routing

The QoS path selection problem has been addressed in previous sections assuming that the *exact* state of the network is known. Such an assumption is often imposed to isolate the impact of network dynamics from the path selection problem. In practice, however, network dynamics can greatly affect the accuracy of the captured and disseminated state information, resulting in some degree of uncertainty in state information.

In current networks, routing protocols are dynamic and distributed. Their dynamic behavior means that important topology changes are flooded to all nodes in the network, while their distributed nature implies that all nodes in the network are equally contributing to the topology information distribution process. Since QoS is associated with resources in the nodes of the network, the QoS link weights are, in general, coupled to these available resources. As illustrated in Figure 8, we distinguish between topology changes that occur infrequently and those that change rapidly. The first kind reflects topology changes due to failures or/and the joining/leaving of nodes. In the current Internet, only this kind of topology changes is considered. Its dynamic is relatively well understood. The key point is that the time between two 'first kind' topology changes is long compared to the time needed to flood this information over the whole network. Thus, the topology databases on which routing relies, converge rapidly with respect to the frequency of updates to the new situation and the transient period where the databases are not synchronized (which may cause routing loops), is generally small.

The second type of rapidly varying changes are typically those related to the consumption of resources or to the traffic flowing through the network. The coupling of the QoS measures to state information seriously complicates the dynamics of flooding because the flooding convergence time T can be longer than the change rate Δ of some metric (such as available bandwidth). Figure 8 illustrates how the bandwidth BW on a link may change as a function of time. In contrast to the first kind changes where $T << \Delta$, in the second kind changes, T can be of the same order as Δ. Apart from this, the second type changes necessitates the definition of a significant change that will trigger the process of flooding. In the first kind, every change was significant enough to start the flooding. The second kind significant change may be influenced by the flooding convergence time T and is, generally, strongly related to the traffic load in (a part of) the network. An optimal update strategy for the second type changes is highly desirable. So far, unfortunately, no optimal topology update rule for

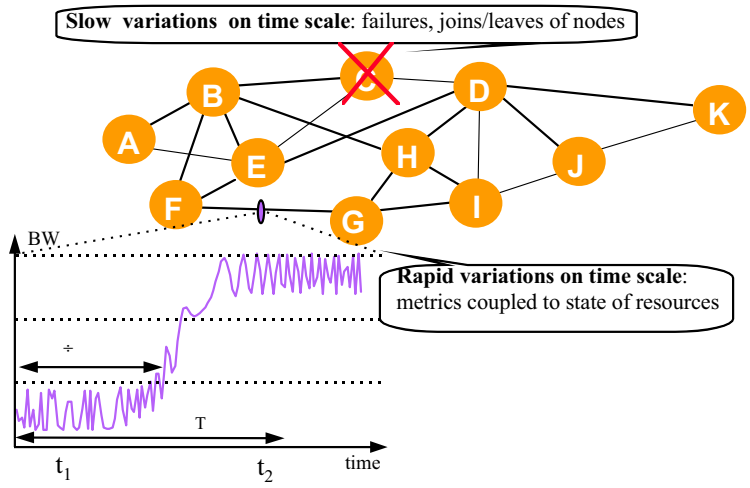


Fig. 8. Network topology changes on different time scales

the second type changes has been published, although some partial results have appeared as outlined in Section 5.

To reduce the overhead of flooding, tree-based broadcasting mechanisms [34] are proposed where a given link state advertisement is delivered only once to every node. Tree-based broadcasting eliminates the unnecessary advertisement overhead, but it introduces a challenging problem, namely how to determine and maintain *consistent* broadcast trees throughout the network. Various tree-based broadcasting mechanisms were proposed for this purpose (e.g., [8,34,9,19]), but they all involve complex algorithms and protocols that cannot be supported with the existing TCP/IP protocol suite. Korkmaz and Krunz [46] proposed a hybrid approach that combines the best features of flooding and tree-based broadcasting.

Besides the update rule (also called triggering policies [56]), a second source of inaccuracy is attributed to state aggregation. Most link-state routing protocols are hierarchical, whereby the state of a group of nodes (an OSPF area or a PNNI peer group) is summarized (aggregated) before being disseminated to other nodes [45,91,89]. While state aggregation is essential to ensuring the scalability of any QoS-aware routing protocol, this information condensation comes at the expense of *perturbing* the true state of the network.

5 Overview of Dynamic QoSR Proposals

A large amount of proposals to deal with the network dynamics are discussed in this section. The multitude of the proposals and the lack of optimal solutions illustrate the challenging difficulty. Moreover, it points to a currently missing functionality in end-to-end quality assured networking.

5.1 Path Selection under Inaccurate Information

As explained in Section 4, some level of uncertainty in state information is unavoidable. To account for such uncertainty, path selection algorithms may follow a *probabilistic* approach in which link state parameters (e.g., delay, available bandwidth) are modelled as random variables (rvs) [28]. Since QoS routing has not yet been implemented in real networks, one of the difficulties lies in what distributions are appropriate for these rvs. In a number of simulation-based studies (e.g., [6,37,38]), a uniformly distributed link bandwidth was assumed while for the link delay, various distributions such as exponential, normal, and gamma were suggested. The exact shape of the distribution may not be a critical issue, as robust path selection algorithms require only knowledge of the statistical moments of the distribution (e.g., *mean and variance*). These statistical moments can be computed simply as follows. Each node maintains a moving average and corresponding variance for a given link state parameter. For example, the moments for the bandwidth can be updated whenever there is a change in the available bandwidth (e.g., flow is added or terminated), while the ones for the delay can be updated whenever a packet leaves the router. In case of a high packet transmission rate, sampling can be used to update the delay parameters. Once the mean and variance are computed for each QoS metric, they can be disseminated using QoS-enhanced versions [5] of OSPF[9]. A crucial question here is when and how to advertise the mean and variance values. A triggered-based approach similar to the one in [3] or [56] can be used for this purpose.

In the case of probabilistically modelled network-state information, the objective of the path selection algorithm is to identify the *most probable* feasible path. This problem has mainly been investigated under bandwidth and/or delay constraints. The general problem at hand can be formulated as follows:

Definition: *Most-Probable Bandwidth-Delay Constrained Path (MP-BDCP) Problem*: Consider a network $G(N, E)$, where N is the set of nodes and E is the set of links. Each link $(i, j) \in E$ is associated with an available bandwidth parameter $b(i, j)$ and a delay parameter $d(i, j)$. It is assumed that the $b(i, j)$'s and $d(i, j)$'s are independent rvs. For any path P from the source node s to the destination node t, let $b(P) \stackrel{\text{def}}{=} \min\{b(i, j) \mid (i, j) \in P\}$ and $d(P) \stackrel{\text{def}}{=} \sum_{(i,j)\in P} d(i, j)$. Given a bandwidth constraint B and a delay constraint D, the problem is to find a path that is most likely to satisfy both constraints. Specifically, the problem is to find a path P^* such that for any other path P from s to t,

$$\pi_B(P^*) \geq \pi_B(P), \quad \text{and} \tag{4}$$

$$\pi_D(P^*) \geq \pi_D(P), \tag{5}$$

where $\pi_B(P) \stackrel{\text{def}}{=} \Pr[b(P) \geq B]$ and $\pi_D(P) \stackrel{\text{def}}{=} \Pr[d(P) \leq D]$.

[9] The current version of OSPF considers only a single, relatively static cost metric. Apostolopoulos *et al.* [5] described a modification to OSPF that allows for disseminating multiple link parameters by exploiting the type-of-service (TOS) field in link-state advertisement (LSA) packets.

If the $b(i,j)$'s and $d(i,j)$'s are constants, the MP-BDCP problem reduces to the familiar bandwidth-delay constrained path problem, which can be easily solved in two steps [95]: (i) prune every link (i,j) for which $b(i,j) < B$, and (ii) find the shortest path w.r.t. the delay parameter in the pruned graph. However, MP-BDCP is, in general, a hard problem. In fact, the objectives (4) and (5) of the MP-BDCP problem give rise to two separate problems: the *most-probable bandwidth constrained path* (MP-BCP) problem and the *most-probable delay constrained path* (MP-DCP) problem. We first review the studies focusing on these problems separately. We then continue our review by considering both parts of the combined MP-BDCP problem simultaneously.

MP-BCP Problem MP-BCP is a rather simple problem, and can be exactly solved by using a standard version of the Most Reliable Path (MRP) algorithm [54,28], which associates a probability measure $\rho(i,j) \stackrel{\text{def}}{=} \Pr[b(i,j) \geq B]$ with every link (i,j). So, $\pi_B(P) = \prod_{(i,j)\in P} \rho(i,j)$. To find a path that maximizes π_B, one can assign the weight $-\log \rho(i,j)$ to each link (i,j) and then run the Dijkstra's shortest path algorithm. In [47] the authors slightly modified the Dijkstra's algorithm for solving the same problem without using logarithms. While the MP-BCP can be efficiently addressed using such exact solutions, the MP-DCP problem is, in general, shown to be NP-hard [25]. Accordingly, most research has focused on the MP-DCP problem.

MP-DCP Problem The MP-DCP problem can be considered under two different models, namely rate-based and delay-based [28]. The "rate-based" model achieves the delay bound by ensuring a minimum service rate to the traffic flow. The main advantage of this model is that the end-to-end delay bound can be mathematically represented depending on the available bandwidth on each link. It seems one can address the MP-DCP problem by using an approach similar to the above MP-BCP problem. In spite of some similarities, however, these problems are not exactly the same due to the fact that the accumulative effect associated with the delay is not produced in the case of bandwidth. In [28] Guerin and Orda showed that the problem is, in general, intractable. Accordingly, they first considered the special cases of the problem and provided tractable solutions for these cases. They then introduced a near-optimal algorithm, named QP, for the MP-DCP problem under rate-based model. Although the rate-based model leads to some attractive solutions, it requires to add new networking mechanisms, mostly regarding using schedulers that allow rate to be strictly guaranteed along the path.

On the other hand, the "delay-based" model provides a general approach for achieving the delay bound by concatenating the local delays associated with each link along the selected path. Note that the above definition formulates the MP-DCP problem based on this general model. The MP-DCP problem is essentially an instance of the stochastic shortest path problem, which has been extensively investigated in the literature (e.g., [58,33]). One key issue in stochastic shortest

path problems, in general, is how to define the optimality of a path. Some formulations (e.g., [66,84,40,77]) aimed at finding the most likely shortest path. Others considered the least-expected-delay paths under interdependent or time-varying probabilistic link delays [85,65,82]. Cheung [15] investigated dynamic stochastic shortest path problems in which the probabilistic link weight is "realized" (i.e., becomes exactly known) once the node is visited. Several studies defined path optimality in terms of maximizing a user-specified objective function (e.g., [58,23,67,69,70]). Our formulation of the MP-DCP problem in the above definition belongs to this category, where the objective is to find a path that is most likely to satisfy the given delay constraint.

Guerin and Orda [28] also considered the MP-DCP problem under the delay-based model and provided tractable solutions for some of its special cases. These cases are relatively limited, so it is desirable to find general tractable solutions which can cope with most network conditions. In [47], Korkmaz and Krunz provided two complementary (approximate) solutions for the MP-DCP problem by employing the central limit theorem approximation and Lagrange relaxation techniques. These solutions were found to be efficient, requiring, on average, a few iterations of Dijkstra's shortest path algorithm. In [28] Guerin and Orda considered a modification of the problem, in which the goal is to partition the given end-to-end delay constraint into local link constraints. The optimal path for the new problem is, in general, different from the one for the MP-DCP problem [59]. Moreover, the solutions provided for the partitioning problem in [59] are computationally more expensive than the solutions in [47] which directly addressed the MP-DCP problem. To reduce the complexity, the authors in [28] have also considered the hierarchical structure of the underlying networks.

Lorenz and Orda have further studied the modified partitioning problem [59]. They first considered the OP (Optimal Partition) Problem and provided an exact solution to it under a particular family of probability distributions (including normal and exponential distributions), where the family selection criterion is based on having a certain convexity property. They then analyzed the OP-MP (Optimally Partitioned Most Probable Path) Problem and provided a pseudo-polynomial solution using dynamic programming methods. In fact, the solution uses a modification of the Dynamic-Restricted Shortest Path Problem (D-RSP). The RSP problem is a well-known problem which aims to find the optimal path that minimizes the cost parameter among all the paths that satisfy the end-to-end delay constraint. Since the RSP Problem is NP-hard, the authors provided a pseudo-polynomial solution from which a new algorithm named Dynamic-OP-MP algorithm is inferred. The main difference between the Dynamic-OP-MP algorithm and the D-RSP algorithm is the cost computation method. As in the OP Problem, the MP-OP Problem is analyzed in detail, particularly when a uniform distribution exists, generating a Uniform-OP-MP algorithm. Finally, they proposed a new approach to obtain a fully polynomial solution to deal with the OP-MP Problem. As in the last case, this approach is based on making some modifications to the D-RSP algorithm, resulting in a non-optimal approximation (named discrete solution). This solution introduces a bounded difference in terms

of cost and success probability regarding the optimal solution by interchanging the cost and delay roles in the D-RSP algorithm.

MP-BDCP Problem MP-BDCP belongs to the class of multi-objective optimization problems, for which a solution may not even exist (i.e., the optimal path w.r.t. π_B is not optimal w.r.t. π_D, or vice versa). To eliminate the potential conflict between the two optimization objectives, one can specify a *utility function* that relates π_B and π_D, and use this function as a basis for optimization. For example, one could maximize $\min\{\pi_B(P), \pi_D(P)\}$ or the product $\pi_B(P)\pi_D(P)$. Rather than optimizing a specific utility function, Korkmaz and Krunz [47] proposed a heuristic algorithm to compute a subset of *nearly nondominated paths* for the given bandwidth and delay constraints. Given this set of paths, a decision maker can select one of these paths according to his/her specific utility function.

5.2 Safety Based Routing

The Safety-Based Routing (SBR) was proposed by Apostolopoulos *et al.* [6]. SBR assumes explicit routing with bandwidth constraints and on-demand path computation. The idea of SBR is to compute the probability that a path can support an incoming bandwidth request. Therefore, SBR computes the Safety (S) parameter defined as the probability that the total required bandwidth is available on the sequence of links that constitute the path. This probability can be used to classify every link, and to find the safest path, i.e. the path having the best chance for supporting total required bandwidth. Since the safety of each link is considered as independent from that of the other links in a path, the safety S of a path is the product of the safeties of every link in that path. Once S has been computed it is included in the path selection process as a new link weight.

SBR uses two different routing algorithms based on combining S with the number of hops, the safest-shortest path and the shortest-safest path. The safest-shortest path algorithm selects that path with the larger safety S among the shortest paths. The shortest-safest path algorithm on the other hand, selects paths with larger safety and if more than one exists the shortest one is chosen. In addition, the SBR mechanism uses triggering policies[10] in order to reduce the signaling overhead while keeping a good routing performance.

A performance evaluation of the blocking probability in [6] showed that the shortest-safest path algorithm is the most effective one for any of the triggering policies that were evaluated.

[10] The most important update policies were discussed and evaluated by Lekovic and Van Mieghem [56]. Similar conclusions were given by Ma, Zhang and Kantola [62]. In addition, they have investigated the performance of update policies under varying network sizes.

5.3 Ticket-Based Distributed QoS Routing

The Ticket-based Distributed QoS Routing mechanism was proposed by Chen and Nahrstedt [13]. They focus on the NP-complete delay-constrained least-cost routing (different from the one explained in Section 2.4). They propose a routing algorithm which targets to find the low-cost path, in terms of satisfying the delay constraint, by using only the available inaccurate or imprecise routing information. To achieve its purpose, initially, Chen and Nahrstedt suggest a simple imprecise state model that defines which information must be stored in every node: connectivity information, delay information, cost information and an additional state variable named delay variation which stands for the estimated maximum change of the delay information before receiving the next updating message. For simplicity reasons, the imprecise model is not applied to the connectivity and cost information. They justify this assumption by saying that the global routing performance is not significantly degraded. Then, a multipath distributed routing scheme, named ticket based probing is proposed. The ticket based probing sends routing messages, named probes, from a source s to a destination d. Based on the (imprecise) network state information available at the intermediate nodes, these probes are routed on a low-cost path that fulfils the delay requirements of the LSP request. Each probe carries at least one ticket in such a way that by limiting the number of tickets, the number of probes is limited as well. Moreover, since each probe searches a path, the number of searched paths is also limited by the number of tickets. In this way, the trade-off between the signalling overhead and the global routing performance may be controlled. Finally, based on this ticket based probing scheme, Chen and Nahrstedt suggest a routing algorithm to address the NP-complete delay-constrained least-cost routing problem, called Ticket Based Probing algorithm (TBP).

Three algorithms were simulated in [13]: the flooding algorithm, the TBP algorithm and the shortest-path algorithm. Simulations are presented using three parameters, the success ratio, the average messages overhead and the average path cost. The results show that the TBP algorithm exhibits a high success ratio and a low-cost path satisfying the delay constraint with minor overhead while tolerating a high degree of inaccuracy in the network state information.

5.4 BYPASS Based Routing

BYPASS based routing (BBR) [63] presented a different idea to solve the bandwidth blocking due to inaccurate routing information produced by a triggering policy based on either threshold based triggers or class based triggers. BBR is an explicit routing mechanism that instructs the source nodes to compute both the working route and as many paths to bypass the links (named bypass-paths) that potentially cannot cope with the incoming traffic requirements. The idea of the BBR mechanism is derived from protection switching for fast rerouting discussed in [14]. However, unlike the use of protection switching for fast rerouting, in BBR both the working and the alternative paths (bypass-paths) are computed simultaneously but not set up; they are only set up when required.

In order to decide those links that might be bypassed (named obstruct-sensitive links, OSLs), a new policy is added. This policy defines a link as OSL whenever a path setup message sent along the explicit route reaches a link with insufficient residual bandwidth. BBR can be implemented using different routing algorithms. Combining the BBR mechanism and Dijkstra's algorithm, two routing algorithms were proposed [63]: the Shortest-Obstruct-Sensitive Path (SOSP), which computes the shortest path among all the paths with the minimum number of obstruct-sensitive links, and the Obstruct-Sensitive-Shortest Path (OSSP), which computes the path that minimizes the number of obstruct-sensitive links among all the shortest paths. In succeeding paper [64], two different routing algorithms were proposed where the residual bandwidth was included in the path selection process: the Widest-Shortest-Obstruct-Sensitive Path (WSOSP) computes the widest path after applying the SOSP algorithm (if more than one feasible path exists) and the Balanced Obstruct-Sensitive Path (BOSP) selects the path that balances network load and network occupancy by modifying the path cost value.

Once the working path is selected, BBR computes the bypass-paths that bypass those links in the working path defined as OSL. When the working path and the bypass-paths are computed, the working path setup process starts. A signaling message is sent along the explicit path included in the setup message. When a node detects that a link in the explicit path has not enough available bandwidth to support the required bandwidth, it sends the setup signaling message over the bypass-path of this link. The bypass-paths nodes are included in the setup signaling message as well, i.e. bypass-paths are also explicitly routed.

The BBR performance is evaluated by simulation in [64]. The obtained results shown in Figure 9 exhibit the reduction obtained in the bandwidth blocking ratio of the BBR (with both SOSP, OSSP, WSOSP and BOSP) compared to the Widest-Shortest Path (WSP) and the Safety Based Routing (Shortest-Safest-Path, SSP). These algorithms are simulated with both the Threshold and the Exponential class triggering policies. Figure 9 indicates that the BOSP algorithm is, in terms of blocking probability, the most effective.

5.5 Path Selection Algorithm Based on Available Bandwidth Estimation

Anjali *et al.* [2] proposed an algorithm for path selection in MPLS networks. They note that most recent QoS routing algorithms utilize the nominal available bandwidth information of the links to optimally select the path. Assuming that most of the traffic flows do not strictly use the requested bandwidth, the nominal link utilization overestimate the actual link consumption, which leads to non-efficient network resource utilization. Therefore, the network performance can be improved by a path selection process based on an accurate measurement of the actual available link bandwidth instead of the nominal value. For scalability reasons, this measurement cannot be reached by any updating link state database process. Moreover, the authors [2] argued that due to the available bandwidth variability, a single sample cannot accurately represent the actual

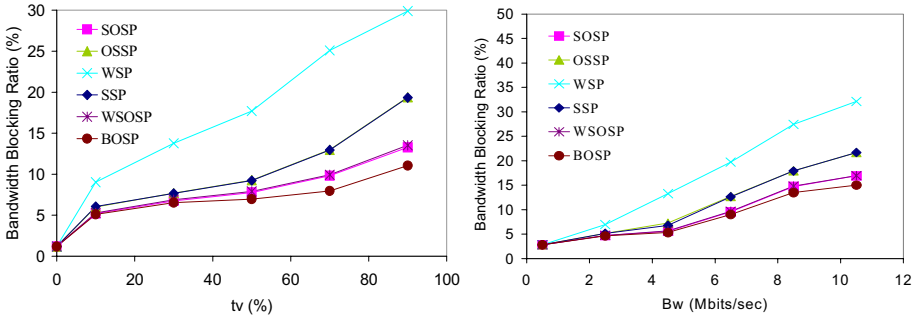


Fig. 9. Bandwidth Blocking Ratio for the threshold and the exponential class triggering policies

bandwidth availability. As a consequence routing decisions based on single samples are likely to be wrong. Since perfectly updated network state information is in general not possible, Anjali *et al.* [2] presented an Available Bandwidth Estimation Algorithm that estimates the actual available bandwidth on each link. A path is computed with a shortest widest path routing algorithm (Section 2.9) that uses these available bandwidth estimations as link weight. Finally, in order to limit the network congestion, a threshold parameter is added. Once the path has been computed, the available bandwidth on the bottleneck link of the path is computed. The threshold parameter is applied to this bottleneck value to compute a benchmark for path selection in such a way that if the bandwidth requested is larger than a certain fraction of the bottleneck link bandwidth, the incoming request is rejected.

The proposed path selection algorithm is shown [2] to perform better than the shortest path routing algorithm in terms of rejection probability, because the proposed routing algorithm based on the available bandwidth estimation algorithm has more accurate information about the actual link load and therefore can take more precise decisions.

5.6 Centralized Server Based QoS Routing

Unlike the previous proposals, Kim and Lee [43] did not attempt to enhance the routing process under inaccurate network state information but rather to eliminate the inaccuracy. Kim and Lee proposed a centralized server based QoS routing scheme, which both eliminates the overhead due to the exchange of network state update messages and achieves higher routing performance by utilizing accurate network state information in the path selection process. Routers are clients of the route server and send routing queries for each one of the incoming requests. The route server stores and maintains two data structures, the Network Topology Data Base (NTDB), which keeps the link state information

for each link in the network, and the Routing Table Cache (RTC) that stores the computed path information.

Although the main idea is derived from that suggested in [4], these new schemes differ in how the network state information is collected. Instead of collecting the link state information from the other routers, in this new approach the proposed router server updates and maintains a link state database as the paths are assigned to or return back from a certain flow. The main issues in this centralized scheme are: (1) the processing load and storage overhead required at the server, (2) the protocol overhead to exchange the router queries and the replies between the server and the remote routers that act as clients and (3) the effects produced when the server becomes either a bottleneck point or a single point of failure. Kim and Lee [43] suggested various alternatives to reduce the loads and overhead.

Two routing algorithms are used: a modification of the Dijkstra's algorithm and the Bellman-Ford algorithm with QoS extensions. Assuming the existence of a certain locality in the communication pattern, a large number of source-destination pairs are expected to be unused. Hence, a path caching approach is used to reduce the path computation overhead. The size of the RTC is controlled by two parameters: the maximum number K of entries (source-destination pairs) in the RTC and the maximum number n of paths for each source-destination pair.

The server based QoS routing scheme was evaluated by simulation [43]. On one hand, the simulations show that a simple path caching scheme substantially reduces the path computation overhead when considering locality in the communication pattern. On the other hand, the simulations indicate that the proposed schemes perform better than the distributed QoS routing schemes with similar protocol overhead.

5.7 A Localized QoS Routing Approach

The main advantage of a localized approach for QoS routing as proposed by Nelakuditi *et al.* [72], is that no global network state information exchange among network nodes is needed, hence reducing the signaling overhead. The path selection is performed in the source nodes based on their local view of the global network state. The main difficulty in implementing any localized QoS routing scheme is how the path is selected only based on the local network state information collected in the source nodes. In order to address this problem Nelakuditi *et al.* present a new adaptive proportional routing approach for localized QoS routing schemes. They propose an idealized proportional routing model, where all paths between a source-destination pair are disjoint and their bottleneck link capacities are known. In addition to this ideal model, the concept of virtual capacity of a path is introduced which provides a mathematically sound way to deal with a shared link among multiple paths. The combinations of these ideas is called Virtual Capacity based routing (VCR). Their simulations [72] showed how the VCR scheme adapts to traffic load changes by adjusting traffic flows to the set of predefined alternative paths. However, Nelakuditi *et al.* described

two significant difficulties related to the VCR implementation that lead them to propose an easily realizable implementation of the VCR scheme, named Proportional Sticky routing (PSR).

The PSR scheme operates in two stages: proportional flow routing and computation of flow proportions. PSR proceeds in cycles of variable lengths. During each cycle, any incoming request can be routed along a certain path selected among a set of eligible paths, which initially may include all the candidate paths. A candidate path is ineligible depending on the maximum permissible flow blocking parameter, which determines how many times this candidate path can block a request before being ineligible. When all candidate paths become ineligible a cycle terminates and all the parameters are reset to start the next cycle. An eligible path is finally selected depending on its flow proportion: the larger the flow proportion, the larger the chances for being selected.

Simulation results show that the PSR scheme is simple, stable and adaptive, and the authors [72] concluded that it is a good alternative to global QoS routing schemes.

5.8 Crankback and Fast Re-routing

Crankback and fast re-routing were included in the ATMF PNNI [7] to address the routing inaccuracy due to fast changes in the resources [90] and due to the information condensation [89] of the hierarchical network structure.

The establishment of a connection between two nodes A and K as shown in Figure 10, takes place in two phases. Based on the network topology reflecting a snap shot at time t_1 and flooded to the last node at $t_1 + T$, the routing algorithm (e.g. SAMCRA) computes the path from A to K subject to some QoS requirements. Subsequently, in the second phase, the needed resources along that path are installed in all nodes constituting that path. This phase is known as the 'connection prerequisite' and the network functionality that reserves resources is called signaling. The signaling operates in a hop by hop mode: it starts with the first node and proceeds further to the next node if the 'installation' is successful. Due to the rapidly changing nature of the traffic in the network, at time $t_2 > t_1$ and at a certain node I (as exemplified in Figure 10), the connection set-up process may fail because the topology situation at time t_1 may significantly differ from that at time t_2 (Figure 8). Rather than immediately blocking the path request from A to K, PNNI invokes an emergency process, called *crankback*. The idea is similar to back tracking. The failure in node I returns the previous node D with the responsibility to compute immediately an alternative path from itself towards K, in the hope that along that new path the set-up will succeed.

The crankback process consumes both much CPU-time in the nodes as control data and yet, does not guarantee a successful connection set-up. When the crankback process returns back to the source node A and this node also fails to find a path to K, the connection request is blocked or rejected and much computational effort of cranking back was in vain.

Although the crankback process seems an interesting emergency solution to prevent blocking, the efficiency certainly needs further study. For, in emergency

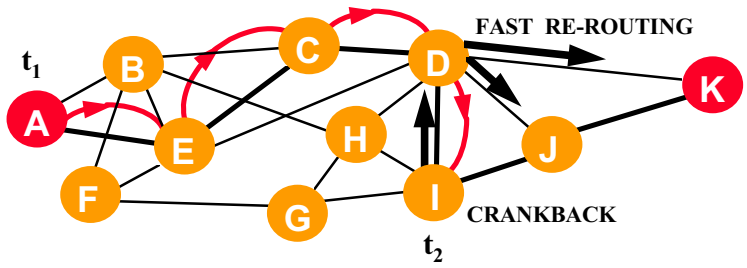


Fig. 10. Illustration of crankback and fast-rerouting.

cases due to heavy traffic, the crankback processes generate additional control traffic possibly causing a triggering of topology flooding, and hence even more control data is created, eventually initiating a positive feedback loop with severe consequences. These arguments suggest to prevent invoking crankback as much as possible by developing a good topology update strategy.

6 Stability Aspects in QoS Routing

If the topology changes as explained in Section 4 are inappropriately fast flooded (and trigger new path computations), route flapping may occur degrading the traffic performance significantly. This section outlines approaches to avoid routing instability.

Routing instabilities were already observed in the ARPANET [42]. The reasons for this routing instability were attributed to the type of link weight sampling used and the path selection algorithm. The use of instantaneous values of the link delay led to frequent changes in the metric, and the shortest paths computed were rapidly out-dated. The application of the Bellman-Ford algorithm with a dynamically varying metric instead of a static metric led to routing loops. These problems were partially overcome by using averaged values of link delay over a ten-second period and by the introduction of a link-state routing protocol as e.g. OSPF. With the constant growth of the Internet, the problem has become recurrent and other solutions had to be found.

6.1 Quantization of QoS Measures and Smoothing

A common approach to avoid routing instability is the advertisement of the link weights that are quantified or smoothed in some manner rather than advertising instantaneous values. This approach has two main consequences, one directly related to routing stability and the other related to routing overhead. The quantization/smoothing limits overshoots in the dynamic metric which reduces the occurrence and the amplitude of routing oscillation. Simultaneously, the distribution of an excessive amount of routing updates is avoided reducing the flooding

overhead. While improving the routing stability, the quantization/smoothing of link weights damps the dynamic coupling to actual resource variations and may lower the adaptation capabilities of the routing protocol. The update strategy consists of a trade-off between routing stability and routing adaptation.

Besides quantization/smoothing of resource coupled link weights, the link weight can be evaluated on different time-scales as proposed by Vutukury and Garcia-Luna-Aceves [92]. A longer time-scale that leads to path computation and a shorter time-scale that allows for the adaptation to traffic bursts.

The techniques of metric quantization/smoothing proposed by Khanna and Zinky [42] reduce routing oscillations, but are not sufficient under adverse circumstances (high loads or bursty traffic) in packet switched networks. When the link weights are distributed, the information may already be out-dated, leading to the typical problem of QoS routing under inaccurate information as discussed in Section 4.

6.2 Algorithms for Load Balancing

Load-balancing provides ways of utilizing multiple-paths between a source and a destination, which may avoid routing oscillations. There are approaches for load balancing in best-effort networks and in QoS-aware networks. Load balancing including QoS can be done per class, per flow or per traffic aggregate (best-effort and QoS flows).

Load Balancing in Best Effort Networks A simple approach of load balancing in best-effort networks is to use alternate paths when congestion rises as in the algorithm Shortest Path First with Emergency Exits (SPF-EE) [94]. This strategy prevents the excessive congestion of the current path because it deviates traffic to an alternate path when congestion starts to rise, and thus avoids routing oscillations. First, the next-hops on the shortest path to all the destinations in the network are determined. Subsequently, the next-hop on the alternate path, called the emergency exit, is added to the routing table. The emergency exit is the first neighbor in the link-state database that is not the next-hop of the shortest path tree nor the final destination. The emergency exit is only used when the queue length exceeds a configured threshold. With this approach two objectives are achieved: the pre-computation of alternate paths allows for traffic distribution over those paths when congestion occurs and the routing update period is increased due to the limitation of traffic fluctuations.

As an alternative to single shortest path algorithms such as SPF-EE, Vutukury and Garcia-Luna-Aceves [92] introduced multiple paths of unequal cost to the same destination. The algorithm proposed by these authors finds near-optimal multiple paths for the same destination based on a delay metric. The algorithm is twofold: it uses information about end-to-end delay to compute multiple paths between each source-destination pair, and local delay information to adjust routing parameters on the previously defined alternate paths. This short scale metric determines the next hop from the list of multiple next-hops that were computed based on the larger scale metric.

Even though the proposals described above permit load balancing and avoid routing oscillations, they do not consider the requirements of the different types of traffic. This problem has been addressed by some proposals within a connection-oriented context.

Load Balancing Supporting QoS Nahrstedt and Chen [71] conceived a combination of routing and scheduling algorithms to address the coexistence of QoS and best-effort traffic flows. In their approach, traffic with QoS guarantees is deviated from paths congested with best-effort traffic in order to guarantee the QoS requirements of QoS flows and to avoid resource starvation of best-effort flows. The paths for QoS flows are computed by a bandwidth-constrained source-routing algorithm and the paths for best-effort flows are computed using max-min fair routing. The authors [71] also addressed the problem of inaccurate information that arises with the use of stale routing information due to the insufficient frequency of routing updates or to dimension of the network. As stated above, inaccurate information is a major contributor to routing instability. To cope with inaccurate information, besides keeping the values of available residual bandwidth (RB) on the link, the estimation on the variation of RB is also kept (ERBV). These two values define the interval (RB-ERBV, RB+ERBV) of the residual bandwidth in the next period. The routing algorithm of QoS flows will find a path between a source and a destination that maximizes the probability of having enough available bandwidth to accommodate the new flow.

Ma and Steenkiste [61] proposed another routing strategy that addresses inter-class resource sharing. The objective of their proposal is also to avoid starvation of best-effort traffic on the presence of QoS flows. The strategy comprises two algorithms: one to route best-effort traffic and the other to route QoS traffic. The routing decisions are based on a metric that enables dynamic bandwidth sharing between traffic classes, particularly, sending QoS traffic through links that are less-congested with best-effort traffic. The metric used for path computation is called virtual residual bandwidth (VRB). The value of the VRB can be above or below the actual residual bandwidth depending on the level of congestion on the link due to best-effort traffic. The algorithm uses the Max-Min Fair Share Rate to evaluate the degree of congestion [41]. If the link is more (less) congested with best-effort traffic than the other links on the network, VRB is smaller (higher) than the actual residual bandwidth. When the link has a small amount of best-effort flows, VRB will be high and the link will be interesting for QoS flows. The paths for best-effort traffic are computed based on the Max-Min Fair Rate for a new connection.

Shaikh *et al.* [80] presented a hybrid approach to QoS routing that takes the characteristics of flows into account to avoid instability. The resources in the network are dynamically shared between short-lived (mice) and long-lived (elephants) flows. The paths for long-lived flows are dynamically chosen, based on the load level in the network, while the paths for short flows are statically pre-computed. Since dynamic routing is only used for long-lived flows, the protocol overhead is limited. At the same time, the duration of these flows avoids successive path computations which is beneficial for stability. The path selection

algorithm computes widest-shortest paths that can accommodate the needs of the flow in terms of bandwidth. This approach is similar to the one used by Vutukury *et al.* described above.

While the above strategies are aimed at connection-oriented networks, the algorithm Enhanced Bandwidth-inversion Shortest-Path [96] was proposed for hop-by-hop QoS routing in Differentiated Services networks. This proposal is based on a Widest-Shortest Path algorithm that takes into account the hopcount. The hopcount is included in the cost function in order to avoid oscillations due to the increased number of flows sent over the widest-path. This approach is similar to the one presented by Shaikh *et al.* [80], but instead of making traffic differentiation per flow, it uses class-based differentiation.

A hop-by-hop QoS routing strategy (UC-QoSR) was developed in [73] for networks where traffic differentiation is class-based. This strategy extends the OSPF routing protocol to dynamically select paths adequate for each traffic class according to a QoS metric that evaluates the impact of the degradation of delay and loss at each router on application performance. The UC-QoSR strategy comprises a set of mechanisms in order to avoid routing instability. Load balancing is embedded in the strategy, since the traffic of all classes is spread over available paths. The link weights are smoothed by using a moving average of its instantaneous values. The prioritizing of routing messages is used to avoid instability due to stale routing information. Combined with these procedures, the UC-QoSR strategy uses a mechanism named class-pinning, that controls the path shifting frequency of all traffic classes. With this mechanism, a new path is used only if significantly better than the path that is currently used by that class [16].

7 Summary and Discussion

Once a suitable *QoS routing protocol* is available and each node in the network has an up to date view of the network, the challenging task in QoS routing is to find a path subject to multiple constraints. The algorithms proposed for the *multi-constrained (optimal) path* problem are discussed and their performance via simulations in the class of Waxman graphs with independent uniformly distributed link weights is evaluated. Table 1 displays the worst-case complexities of the algorithms discussed in Section 2.

The simulation results show that the worst-case complexities of Table 1 should be interpreted with care. For instance, the actual execution time of H_MCOP will always be longer than that of Jaffe's algorithm under the same conditions. In general, the simulation results indicate that TAMCRA-like algorithms that use a k-shortest path algorithm and a nonlinear length function while eliminating dominated paths and possibly applying other search-space reducing techniques such as look-ahead perform best. The performance and complexity of TAMCRA-like algorithms is easily adjusted by controlling the value of k. When k is not restricted, TAMCRA-like algorithms as SAMCRA lead to exact solutions. In the class of Waxman or random graphs with uniformly distributed link

Algorithm	Worst-case complexity
Jaffe's algorithm	$O(N \log N + mE)$
Iwata's algorithm	$O(mN \log N + mE)$
SAMCRA, TAMCRA	$O(kN \log(kN) + k^2 mE)$
EDSP, EBF	$O(x_2^2 \cdots x_m^2 N^2)$, $O(x_2 \cdots x_m NE)$
Randomized algorithm	$O(mN \log N + mE)$
H_MCOP	$O(N \log N + mE)$
LPH	$O(k^2 NE)$
A*Prune	$O(QN(m + N + \log h))$

Table 1. Worst-case complexities of QoS routing algorithms.

weights, simulations suggest that the execution times of such exact algorithms increase almost linearly with the number of nodes in $G(N, E)$, contrary to the expected exponential (NP) increase.

The study reveals that the exact algorithm SAMCRA (and likewise TAMCRA) can be extended with the look-ahead property. The combination of the four powerful concepts (non-linear definition of length, k-shortest paths, dominance and look-ahead) into one algorithm makes SAMCRAv2 the current most efficient exact QoS routing algorithm.

The second part of this chapter has discussed the dynamics of QoS routing. Mainly QoS routing without complete topology information and the stability of QoS routing are addressed. A probabilistic approach to incorporate the complex dynamic network processes is reviewed. While the study of QoS routing algorithms has received due attention, the routing dynamics and the behavior of the QoS routing protocol deserve increased efforts because these complex processes are insufficiently understood. Several proposals are outlined in Section 5, but the performance of these proposal has not been compared yet. As a result, a commonly accepted QoS routing protocol is a still missing functionality in today's communication networks.

List of Open Issues

At the time of writing, we believe that the following open problems deserve to be placed on the agenda for future research:

- Determining for which graphs and link weight structures the MC(O)P is not NP-complete.
- A detailed and fair comparison of the proposed dynamic aspects of QoS routing proposals. Usually, authors propose an idea and choose a few simulations to show the superiority of their approach compared to other proposals.
- Designing efficient QoS routing protocols.
- Aiming for an optimized QoS routing protocol.
- The deployment of QoS routing for Diffserv.
- Combined approaches of QoS routing and QoS signaling.
- QoS multicast routing.

– QoS routing implications on layer 2 technologies.
– QoS routing in Adhoc networks and peer-to-peer networks

References

1. L.H. Andrew and A.A.N. Kusuma, "Generalized analysis of a qos-aware routing algorithm," Proc. of IEEE GLOBECOM 1998, Piscataway, NJ, USA, vol. 1, pp. 1-6, 1998.
2. T. Anjali, C. Scoglio, J. de Oliveira, L.C. Chen, I.F. Akyldiz, J.A. Smith, G. Uhl, A. Sciuto, "A new path selection algorithm for MPLS networks based on available bandwidth estimation," Proc. of QofIS 2002, pp.205-214, Zurich, Switzerland, October 2002.
3. G. Apostolopoulos, R. Guerin, S. Kamat and S.K. Tripathi, "Quality of service based routing: a performance perspective," Proc. of ACM SIGCOMM '98, Vancouver, British Columbia, Canada, pp. 17-28, August/September, 1998.
4. G. Apostolopoulos, R. Guerin, S. Kamat and S.K. Tripathi, "Server based qos routing," Proc. of IEEE GLOBECOM 1999, 1999.
5. G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda and T. Przygienda, "QoS routing mechanisms and OSPF extensions," RFC 2676, August 1999.
6. G. Apostolopoulos, R. Guerin, S. Kamat, and S. Tripathi, "Improving qos routing performance under inaccurate link state information," Proc. of the 16th International Teletraffic Congress (ITC '16), Edinburgh, United Kingdom, June 7-11, 1999.
7. The ATM Forum, "Private network-to-network interface specification version 1.1 (PNNI 1.1)," af-pnni-0055.002, April 2002.
8. E. Basturk and P. Stirpe, "A hybrid spanning tree algorithm for efficient topology distribution in PNNI," Proc. of the 1st IEEE International Conference on ATM (ICATM '98), pages 385–394, 1998.
9. B. Bellur and R.G. Ogier, "A reliable, efficient topology broadcast protocol for dynamic networks," IEEE INFOCOM'99, volume 1, pages 178-186, 1999.
10. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss, "An architecture for differentiated services," RFC 2475, December 1998.
11. R. Braden, D. Clark and S. Shenker, "Integrated services in the internet architecture: an overview," RFC 1633, June 1994.
12. S. Chen and K. Nahrstedt, "On finding multi-constrained paths," Proc. of ICC '98, New York, pp. 874-879, 1998.
13. S. Chen and K. Nahrstedt, "Distributed qos routing with imprecise state information," Proc. of 7th IEEE International Conference of Computer, Communications and Networks, Lafayette, LA, pp. 614-621, October 1998.
14. T.M. Chen and T.H. Oh, "Reliable services in MPLS," IEEE Communications Magazine, pp. 58-62, 1999.
15. R.K. Cheung, "Iterative methods for dynamic stochastic shortest path problems," Naval Research Logistics, no. 45, pp. 769–789, 1998.
16. M. Curado, O. Reis, J. Brito, G. Quadros and E. Monteiro, "Stability and scalability issues in hop-by-hop class-based routing," Proc. of the 2nd International Workshop on QoS in Multiservice IP Networks (QoS-IP2003), Milano, Italy, February 24-26, 2003.
17. E.I. Chong, S. Maddila and S. Morley, "On finding single-source single-destination k shortest paths," J. Computing and Information, special issue ICCI'95, pp. 40-47, 1995.

18. T. H. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, MIT Press, Cambridge, MA, 2000.
19. Y.K. Dalal and R.M. Metcalfe, "Reverse path forwarding of broadcast packets," Communications of the ACM, no. 21, pp. 1040-1048, December 1978.
20. H. De Neve and P. Van Mieghem, "A multiple quality of service routing algorithm for PNNI," IEEE ATM workshop, Fairfax, pp. 324-328, May 26-29, 1998.
21. H. De Neve and P. Van Mieghem, "TAMCRA: a tunable accuracy multiple constraints routing algorithm," Computer Communications, vol. 23, pp. 667-679, 2000.
22. E. W. Dijkstra, "A note on two problems in connexion with graphs," Numerische Mathematik, no. 1, pp. 269-271, 1959.
23. A. Eiger, P.B. Mirchandani, and H. Soroush, "Path preferences and optimal paths in probabilistic networks," Transportation Science, vol. 19, no. 1, pp. 75–84, February 1985.
24. B. Fortz and M. Thorup, "Internet traffic engineering by optimizing OSPF weights," IEEE INFOCOM 2000, vol. 2, pp. 519-528, 2000.
25. H. Frank, "Shortest paths in probabilistic graphs," Oper. Res., vol. 17, 583–599, 1969.
26. M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*, Freeman, San Francisco, 1979.
27. G.H. Golub and C.F. Van Loan, *Matrix Computations*, 1st ed., North Oxford Academic, Oxford, 1983.
28. R. Guerin and A. Orda, "QoS routing in networks with inaccurate information: theory and algorithms," IEEE/ACM Transactions on Networking, vol. 7, no. 3, pp. 350-364, June 1999.
29. R. Guerin and A. Orda, "Networks with advance reservations: the routing perspective," IEEE INFOCOM 2000, Israel, March 26-30, 2000.
30. L. Guo and I. Matta, "Search space reduction in qos routing", Proc. of the 19th III Int. Conference on Distributed Computing Systems, III, pp. 142-149, May 1999.
31. Y. Guo, F. A. Kuipers and P. Van Mieghem, 2003, "A Link-Disjoint Paths Algorithm for Reliable QoS Routing", to appear in International Journal of Communication Systems.
32. R. Hassin, "Approximation schemes for the restricted shortest path problem," Mathematics of Operations Research, vol. 17, no. 1, pp. 36–42, 1992.
33. M.I. Henig, "The shortest path problem with two objective functions," European J. of Operational Research, vol. 25, pp. 281-291, 1985.
34. P.A. Humblet and S.R. Soloway, "Topology broadcast algorithms," Computer Networks and ISDN Systems, vol. 16, 179-186, 1988/89.
35. A. Iwata, R. Izmailov, D.-S. Lee, B. Sengupta, G. Ramamurthy and H. Suzuki, "ATM routing algorithms with multiple qos requirements for multimedia internetworking," IEICE Transactions and Communications E79-B, no. 8, pp. 999-1006, 1996.
36. J.M. Jaffe, "Algorithms for finding paths with multiple constraints," Networks, no. 14, pp. 95-116, 1984.
37. Y. Jia, I. Nikolaidis and P. Gburzynski, "Multiple path routing in networks with inaccurate link state information," IEEE ICC, vol. 8, pp. 2583–2587, 2001.
38. W. Jianxin, W. Weiping, C. Jianer and C. Songqiao, "A randomized qos routing algorithm on networks with inaccurate link-state information," Proc. of the International Conference on Communication Technology (WCC - ICCT 2000), vol. 2, pp. 1617-1622, 2000.

39. A. Juttner, B. Szviatovszki, I. Mecs and Z. Rajko, "Lagrange relaxation based method for the qos routing problem," IEEE INFOCOM 2001, vol. 2, pp. 859–868, April 2001.
40. J. Kamburowski, "A note on the stochastic shortest route problem," Operations Research, vol. 33, no. 3, pp. 696–698, May-June 1985.
41. S. Keshav, *An Engineering Approach to Computer Networking: ATM networks, the Internet, and the Telephone Network*, Addison-Wesley, 1997.
42. A. Khanna and J. Zinky, "The revised ARPANET routing metric," SIGCOMM'89, 1989.
43. S. Kim and M. Lee, "Server based qos routing with implicit network state updates," IEEE GLOBECOM 2001, vol.4, pp. 2182-2187, San Antonio, Texas, November 2001.
44. M. Kodialam and T.V. Lakshman, "Dynamic routing of bandwidth guaranteed tunnels with restoration," IEEE INFOCOM 2000, pp. 902-911, 2000.
45. T. Korkmaz and M. Krunz, "Source-oriented topology aggregation with multiple qos parameters in hierarchical networks," The ACM Transactions on Modeling and Computer Simulation (TOMACS), vol. 10, no. 4, pp. 295–325, October 2000.
46. T. Korkmaz and M. Krunz, "Hybrid flooding and tree-based broadcasting for reliable and efficient link-state dissemination," IEEE GLOBECOM '02 Conference - High-Speed Networks Symposium, November 2002.
47. T. Korkmaz and M. Krunz, "Bandwidth-delay constrained path selection under inaccurate state information," to appear in IEEE/ACM Transactions on Networking, 2003.
48. T. Korkmaz and M. Krunz, "A randomized algorithm for finding a path subject to multiple qos requirements," Computer Networks, vol. 36, pp. 251-268, 2001.
49. T. Korkmaz and M. Krunz, "Multi-constrained optimal path selection," IEEE INFOCOM, 2001.
50. F.A. Kuipers and P. Van Mieghem, "QoS routing: average complexity and hopcount in m dimensions," Proc. of Second COST 263 International Workshop, QofIS 2001, Coimbra, Portugal, pp. 110-126, September 24-26, 2001.
51. F.A. Kuipers and P. Van Mieghem, "MAMCRA: a constrained-based multicast routing algorithm," Computer Communications, vol. 25/8, pp. 801-810, May 2002.
52. F. A. Kuipers and P. Van Mieghem, "The impact of correlated link weights on qos routing", IEEE INFOCOM, San Francisco, USA, April 2003.
53. F.A. Kuipers and P. Van Mieghem, "Bi-directional search in qos routing," accepted to 4th Cost 263 International Workshop on Quality of Future Internet Services (QofIS 2003), Stockholm, Sweden, October 1-3, 2003.
54. E.L. Lawler, *Combinatorial Optimization: networks and matroids*, New York: Holt, Rinehart and Winston, 1976.
55. W.C. Lee, M.G. Hluchyi and P.A. Humblet, "Routing subject to quality of service constraints in integrated communication networks," IEEE Network, pp. 46-55, July/August, 1995.
56. B. Lekovic and P. Van Mieghem, "Link state update policies for quality of service routing," IEEE Eighth Symposium on Communications and Vehicular Technology in the Benelux (SCVT2001), Delft, The Netherlands, pp. 123-128, October 18, 2001.
57. G. Liu and K.G. Ramakrishnan, "A*Prune: an algorithm for finding K shortest paths subject to multiple constraints," IEEE INFOCOM, 2001.
58. R.P. Loui, "Optimal paths in graphs with stochastic or multidimensional weights," Communications of ACM, vol. 26, no. 9, pp. 670–676, 1983.

59. D.H. Lorenz and A.Orda, "QoS routing in networks with uncertain parameters," IEEE/ACM Transactions on Networking, vol.6, no. 6, pp. 768-778, December 1998.
60. Q. Ma and P. Steenkiste, "Quality-of-service routing with performance guarantees," Proc. of 4th Int. IFIP Workshop on QoS, May 1997.
61. Q. Ma and P. Steenkiste, "Supporting dynamic inter-class resource sharing: a multi-class qos routing algorithm," IEEE INFOCOM, 1999.
62. Z. Ma, P. Zhang and R. Kantola, "Influence of link state updating on the performance and cost of qos routing in an intranet," 2001 IEEE Workshop on High Performance Switching and Routing (HPSR 2001), Dallas, Texas USA, May 29-31, 2001.
63. X.Masip-Bruin, S.Sánchez-López, J.Solé-Pareta, J.Domingo-Pascual, "A qos routing mechanism for reducing inaccuracy effects," Proc. of QoS-IP, Milán, Italy, February 2003.
64. X.Masip-Bruin, S.Sánchez-López, J.Solé-Pareta, J.Domingo-Pascual, "QoS routing algorithms under inaccurate routing information for bandwidth constrained applications, " Proceedings of International Communications Conference, IEEE ICC'03, Anchorage, Alaska, May 2003.
65. E.D. Miller-Hooks and H.S. Mahmassani, "Least expected time paths in stochastic, time-varying transportation networks," Transportation Science, vol. 34, no. 2, pp. 198-215, May 2000.
66. P.B. Mirchandani, "Shortest distance and reliability of probabilistic networks," Comput. & Ops. Res., vol. 3, pp. 347–355, 1976.
67. P.B. Mirchandani and H. Soroush, "Optimal paths in probabilistic networks: a case with temporal preferences," Comput. and Operations Research, vol. 12, no. 4, pp. 365–381, 1985.
68. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
69. I. Murthy and S. Sarkar, "Exact algorithms for the stochastic shortest path problem with a decreasing deadline utility function," European Journal of Operational Research, vol. 103, pp. 209–229, 1997.
70. I. Murthy and S. Sarkar, "Stochastic shortest path problems with piecewise-linear concave utility functions," Management Science, vol. 44, no. 11, pp. S125—S136, Nov. 1998.
71. K. Nahrstedt and S. Chen, "Coexistence of qos and best effort flows - routing and scheduling," Proc. of 10th IEEE Tyrrhenian International Workshop on Digital Communications: Multimedia Communications, Ischia, Italy, September, 1998.
72. S. Nelakuditi, Z. Zhang and R.P. Tsang, "Adaptive proportional routing: a localized qos routing approach," IEEE INFOCOM 2000, pp. 1566-1575, 2000.
73. M. Oliveira, J. Brito, B. Melo, G. Quadros and E. Monteiro, "Quality of service routing in the differentiated services framework," Proc. of SPIE's International Symposium on Voice, Video, and Data Communications (Internet III: Quality of Service and Future Directions), Boston, Massachusetts, USA, November 5-8, 2000.
74. A. Orda, "Routing with end-to-end qos guarantees in broadband networks," IEEE/ACM Transactions on Networking, vol. 7, no. 3, pp. 365-374, 1999.
75. A. Orda and A. Sprintson, "QoS routing: the precomputation perspective," IEEE INFOCOM 2000, pp. 128-136, 2000.
76. M. Peyravian and A.D. Kshemkalyani, "Network path caching: issues, algorithms and a simulation study," Performance Evaluation, vol. 20, no. 8, pp. 605-614, 1997.
77. G.H. Polychronopoulos and J.N. Tsitsiklis, "Stochastic shortest path problems with recourse," Operations Research Letters, vol. 10, pp. 329–334, August 1991.

78. D.S. Reeves and H.F. Salama, "A distributed algorithm for delay-constrained unicast routing," IEEE/ACM Transactions on Networking, vol. 8, no. 2, pp. 239–250, April 2000.
79. E. Rosen, A. Viswanathan and R. Callon, "Multiprotocol label switching architecture," RFC 3031, January 2001.
80. A. Shaikh, J. Rexford and K. Shin, "Load-sensitive routing of long-lived IP flows," ACM SIGCOMM'99, 1999.
81. H.F. Salama, D.S. Reeves and Y. Viniotis, "Evaluation of multicast routing algorithms for real-time communication on high-speed networks," IEEE JSAC, vol. 15, no. 3, pp. 332-345, April 1997.
82. S. Sen, R. Pillai, S. Joshi and A.K. Rathi, "A mean-variance model for route guidance in advanced traveler information systems," Transportation Science, vol. 35, no. 1, pp. 37–49, February 2001.
83. A. Schrijver, *Combinatorial Optimization*, Vol. A - Vol. C, Springer-Verlag, Berlin, 2003
84. C.E. Sigal, A.A.B. Pritsker and J.J. Solberg, "Stochastic shortest route problem," Operations Research, vol. 28, no. 5, pp. 1122-1129, Sept.-Oct. 1980.
85. R.A. Sivakumar and R. Batta, "The variance-constrained shortest path problem," Transportation Science, vol. 28, no. 4, pp. 309–316, Nov. 1994.
86. N. Taft-Plotkin, B. Bellur and R. Ogier, "Quality-of-service routing using maximally disjoint paths," Proc. of the Seventh International Workshop on Quality of Service (IWQoS'99), London, England, pp. 119-128, May/June, 1999.
87. P. Van Mieghem, H. De Neve and F.A. Kuipers, "Hop-by-hop quality of service routing," Computer Networks, vol. 37, no. 3-4, pp. 407-423, October 2001.
88. P. Van Mieghem, "Paths in the simple random graph and the Waxman graph," Probability in the Engineering and Informational Sciences (PEIS), vol. 15, pp. 535-555, 2001.
89. P. Van Mieghem, "Topology information condensation in hierarchical networks," Computer Networks, vol.. 31, no. 20, pp. 2115-2137, November, 1999.
90. P. Van Mieghem and H. De Neve, "Aspects of quality of service routing," Proc. of SPIE'98, Boston (USA), 3529A-05, Nov. 1-6, 1998.
91. P. Van Mieghem, "Estimation of an optimal PNNI topology," IEEE ATM Workshop, pp. 570–577, 1997.
92. S. Vutukury and J.J. Garcia-Luna-Aceves, "A simple approximation to minimum-delay routing," ACM SIGCOMM'99, 1999.
93. B. Wang and J.C. Hou, "Multicast routing and its qos extension: problems, algorithms, and protocols," IEEE Network, vol. 14, no. 1, pp. 22-36, Jan.-Feb 2000.
94. Z. Wang and J. Crowcroft, "Shortest path first with emergency exits," SIGCOMM'90, Philadelphia, USA, September 1990.
95. Z. Wang and J. Crowcroft, "Quality-of-service routing for supporting multimedia applications," IEEE JSAC, vol. 14, no. 7, pp. 1228-1234, September, 1996.
96. J. Wang and K. Nahrstedt, "Hop-by-hop routing algorithms for premium-class traffic in diffserv networks," IEEE INFOCOM, 2002.
97. B.M. Waxman, "Routing of multipoint connections," IEEE JSAC, vol. 6, no. 9, pp. 1617-1622, December 1998.
98. X. Xiao and L.M. Ni, "Internet qos: a big picture," IEEE Network, vol. 13, no. 2, pp. 8-18, March-April 1999.
99. X. Yuan, "Heuristic algorithms for multiconstrained quality-of-service routing," IEEE/ACM Transactions on Networking, vol. 10, no. 2, April 2002.

Internet Traffic Engineering

O. Bonaventure[1], P. Trimintzios[2], G. Pavlou[2], B. Quoitin[3] [Editors],
A. Azcorra[4], M. Bagnulo[4], P. Flegkas[2], A. Garcia-Martinez[4], P. Georgatsos[5],
L. Georgiadis[6], C. Jacquenet[7], L. Swinnen[3], S. Tandel[3], and S. Uhlig[1]

[1] CSE Dept, Université Catholique de Louvain, Belgium
[2] Centre for Communication Systems Research,
University of Surrey, Guildford, Surrey, GU2 7XH, U.K.
[3] Infonet group, University of Namur, Belgium
[4] Departamento de Ingeniera Telemtica,
Universidad Carlos III, Madrid, Spain
[5] Algonet S.A. 206 Syggrou Ave, 17 672, Athens, Greece
[6] Electrical and Computer Engineering Dept.
Aristotle University of Thessaloniki, 54 124, Thessaloniki, Greece
[7] France Telecom, Rennes, France

Abstract. Traffic engineering encompasses a set of techniques that can be used to control the flow of traffic in data networks. We discuss several of those techniques that have been developed during the last few years. Some techniques are focused on pure IP networks while others have been designed with emerging technologies for scalable Quality of Service (QoS) such as Differentiated Services and MPLS in mind. We first discuss traffic engineering techniques inside a single domain. We show that by using a non-linear programming formulation of the traffic engineering problem it is possible to meet the requirements of demanding customer traffic, while optimising the use of network resources, through the means of an automated provisioning system. We also extend the functionality of the traffic engineering system through policies. In the following, we discuss the techniques that can be used to control the flow of packets between domains. First, we briefly describe interdomain routing and the Border Gateway Protocol (BGP). Second, we summarise the characteristics of interdomain traffic based on measurements with two different Internet Service Providers. We show by simulations the limitations of several BGP-based traffic engineering techniques that are currently used on the Internet. Then, we discuss the utilisation of BGP to exchange QoS information between domains by using the QOS_NLRI attribute to allow BGP to select more optimum paths. Finally, we consider the multi-homing problem and analyse the current proposed IPv6 multi-homing solutions are analysed along with their impact on communication quality.

1 Introduction

The Internet is becoming a targeted support for a wide range of IP service offerings, ranging from dial-up access to more sophisticated offers, such as Virtual Private Networks. The success of some of these services is now conditioned by

M. Smirnov et al. (Eds.): COST 263 Final Report, LNCS 2856, pp. 118–179, 2003.

the ability of the service providers to commit on the provisioning of a guaranteed level of quality, which will possibly be negotiated with the customer, and which will depend on the network resource availability.

Internet Traffic Engineering (TE) is one of the methods that can be used by service providers to commit to those guarantees. TE is defined as that aspect of Internet network engineering dealing with the issue of performance evaluation and performance optimisation of operational IP networks [1]. Different techniques are applicable for isolated domains, also called autonomous systems (AS) and for the global Internet which is composed of more than 13.000 distinct ASes. This chapter discusses both these applications of traffic engineering.

Inside a single domain, Traffic Engineering systems can be categorised based on the different TE styles and views [1]. Time-dependent TE uses historical information based on periodic variations in traffic to pre-program routing plans and other TE control mechanisms, while state-dependent TE adapts dynamically the routing plans based on the current state of the network. The route computation can be done either on-line or off-line. In centralised TE a central authority determines the routing plans and other TE control parameters, while in distributed TE route selection is determined by each router autonomously, based on the router's view of the state of the network.

Differentiated Services (DiffServ) [2] is the emerging technology to support Quality of Service (QoS) in IP backbone networks in a scalable fashion. Multi-Protocol Label Switching (MPLS) [3] can be used as the underlying technology to support traffic engineering. It is possible to use these technologies in conjunction in order to provide adequate quality guarantees for traffic with QoS requirements. This can be done through careful traffic forecasting based on contracted services with customers and subsequent network provisioning in terms of routing and resource provisioning. In the first part of this chapter we discuss how to meet traffic requirements while optimising the use of the intra-domain resources with a traffic engineering system. In addition we use the policy-based management paradigm [4] to dynamically guide the behaviour of the traffic engineering system that provisions the network in order to meet high-level business objectives.

Besides the need to optimise the flow of packets inside a network, there is also a strong need to optimise the flow of packets between networks. At the time of writing, MPLS is not yet used to solve traffic engineering purposes between domains. Thus the only solution today is to rely on the existing interdomain routing protocol, namely the Border Gateway Protocol [5].

In the second part of this chapter, we focus on four issues related to the support of traffic engineering across interdomain boundaries. The first issue has to do with the characteristics of the interdomain traffic that we summarise based on measurements taken from several ISPs. A second issue is how BGP can be used to control the flow of best-effort interdomain traffic. We highlight the limitations of the current BGP-based traffic engineering techniques through simulations. However, as mentioned earlier, Differentiated Services are more and more used inside isolated autonomous systems and there is a clear need to support similar services across interdomain boundaries. One of the ways to provide such

services is by extending BGP to distribute QoS information. The last issue that we address is that autonomous systems are often multi-homed, i.e. connected to several providers. This multi-homing is often used to improve the performance or reduce the cost of the interdomain traffic. However, this multi-homing also stresses the interdomain routing system by increasing the size of the BGP routing tables. Better multi-homing strategies are currently being developed for IPv6.

This chapter is divided in two main parts. The first part focuses on intra-domain traffic engineering and begins with Sect. 2 that presents the related work. Then, Sect. 3 describes a Traffic Engineering System Architecture. Section 4 describes the system's operation cycle and in Sect. 5 we present an algorithm for network dimensioning, i.e. offline traffic engineering, with some simulation results. In Sect. 6 we enlist the potential policies related to network dimensioning and we present a policy enforcement example.

The second part of the chapter is devoted to the issues that arise when considering the interconnection of distinct domains. In section 7 we briefly describe interdomain routing and the BGP protocol. Then, in section 8, we summarise the characteristics of interdomain traffic. In section 9.1, we show the difficulty of selecting Internet paths based on the current BGP attributes by studying BGP routing tables from various ISPs. In section 9.2, we present a detailed simulation study of the performance of one technique often used by ISPs to control their incoming traffic. Section 10 is devoted to the discussion of QoS extensions to BGP. Finally, in section 11 the discuss several approaches to the multi-homing problem in both IPv4 and IPv6 environments.

2 Related Work in Intra-domain Traffic Engineering

The problem of traffic engineering has attracted a lot of attention in recent years. Traffic Engineering entails the aspect of network engineering that is concerned with the design, provisioning, and tuning of operational Internet networks. In order to deal with this important emerging area, the Internet Engineering Task Force (IETF) has chartered the Internet Traffic Engineering Working Group (tewg) [6] to define, develop, specify, and recommend principles, techniques and mechanisms for traffic engineering in IP-based networks. The IETF has defined the basic principles for traffic engineering [1], the requirements for MPLS traffic engineering [7], and the requirements to support the inter-operation of MPLS and Diffserv for traffic engineering [8]. It is in the plans of tewg to look into technical solutions for meeting the requirements for Diffserv-aware MPLS traffic engineering, the necessary protocol extensions, inter-operability proposals and measurement requirements. In addition there some recent proposals in the IETF to extend the information included in Link State Advertisements (LSAs) of intra-domain routing protocols, like Open Shortest Path First (OSPF). The traffic engineering extensions [9] to OSPF, will enable the flooding of the extended LSAs to all routers within a domain, which will then store the received TE information into a TE Database (TED) so that it can be facilitated by constraint path computation algorithms.

Two similar works with the work presented here are the Netscope [10] and RATES [11]. Both of them try to automate the configuration of the network in order to maximise network utilisation. The first one uses measurements to derive the traffic demands and then by employing the offline algorithm described in [12] it tries to offload overloaded links. The latter uses the semi-online algorithm described in [13] to find the critical links which if they are chosen for routing will cause the greatest interference (i.e. reduce the maximum flow) of the other egress-ingress pairs of the network. Both of these works do not take into account any QoS requirements and only try to minimise the maximum load of certain links.

The traffic engineering and provisioning work we will describe in the following sections is aimed to be used in conjunction with service management functionalities [14], and all together could be parts of an extended Bandwidth Broker (BBs) [15]. A BB is an agent that works within a domain, keeps track of the domain's resource allocation and accepts or rejects new requests for using the network [15]. The extended BB could in addition provision the network, and thus by creating the resource allocation is in position to take more flexible decisions. Another functionality to which a BB can play an intermediate role is that of end-to-end allocation of resources, built through the peering connections with the adjacent domain's BBs. There were recently some proposals for such end-to-end frameworks [16] and [17]. The work in [16] proposes a two level admission control, one for the provisioning of QoS-pipes between and within domains, and a second level for individual flows which make use of the pre-provisioned QoS-pipes. This two level approach to admission control, proposed also in [18], fits very well with our two-level traffic engineering system and thus could be used in conjunction, to form parts of a BB. This chapter will focus on the traffic engineering and provisioning functionalities of the BB, a broad discussion on the service engineering functionalities can be found in [19].

The offline traffic engineering and provisioning, which we collectively call *network dimensioning*, algorithm described later in this chapter targets to solve problems that can be categorised as (class-based) *offline* traffic engineering [1]. Such problems can be naturally modelled as multi-commodity network flow optimisation problems [20]. The related works use optimisation formulations, focusing on the use of linear cost functions, usually the sum of bandwidth requirements, and in most of the cases they try to optimise a single criterion, i.e. minimise the total network cost.

The advantage of the linear problem formulation is that it can be optimally solved by using linear programming methods, i.e. the network simplex algorithm [20]. On the other hand, a linear cost function of link load does not penalise heavily-loaded links enough, resulting in poorer traffic distribution across the network. In addition, such linear formulations can take into account more than one optimisation criteria as a linear combination of the respective objectives which is not very flexible. In our approach presented here, we formulate the problem in a non-linear fashion, combining as criteria the minimisation of both total network cost and of maximum link load.

In [21] the traffic-engineering problem is seen as a multi-priority problem, formulated as a multi-criterion optimisation problem on a predefined traffic matrix. This approach uses the notion of predefined admissible routes that are specific for each QoS class and each source-destination pair, where the objective is the maximisation of the carried bandwidth. In [22], the authors address the resource allocation and routing problem in the design of Virtual Private Networks (VPNs). The main objective is to design VPNs which will have allocated bandwidth on the links of the infrastructure network such that, when the traffic of a customer is optimally routed, a weighted aggregate measure over the service provider's infrastructure is maximised, subject to the constraint that each VPN carries a specified minimum. The weighted measure is the network revenue, which is a function of the traffic intensity. The algorithm proposed in that paper solves first the optimal routing problem for each VPN independently. Then it calculates for each VPN the linear capacity costs for all the links. These quantities are used to modify appropriately the current capacity allocations so that the network revenue of the infrastructure network for the new capacities is maximised. It is shown in [22] that this is equivalent to minimising a linear function of the capacity costs subject to constraints imposed by the link capacities.

In [23] a model is proposed for off-line centralised traffic engineering over MPLS. This uses one of the following objectives: resource-oriented or traffic-oriented traffic engineering. The resource-oriented problem targets load balancing and minimisation of resource usage. Capacity usage is defined as the total amount of capacity used and load balancing is defined as one minus the maximal link utilisation. The objective function that has to be maximised is a linear combination of capacity usage and load balancing, subject to constraints imposed by the capacity of the links. The traffic-oriented model suggests an objective function that is a linear combination of fairness and throughput, where throughput is defined as the total bandwidth guaranteed by the network and fairness as the minimum weighted capacity allocated to a traffic trunk. In [24] the authors propose an algorithm which has two phases, a pre- processing phase and an on-line one. In the pre-processing phase the algorithm uses the notion of multi-commodity flows, where commodities correspond to traffic classes. The goal is to find paths in the network to accommodate as much traffic as possible from the source to the destination node. The algorithm tries to minimise a linear cost function of the bandwidth assigned to each link for a traffic class. The second phase performs the on-line path selection for LSP requests by using the pre-computed output of the multi-commodity pre-processing phase.

The offline traffic engineering works discussed so far, are assuming that the anticipated traffic is estimated in the form of a traffic matrix on ingress to egress node basis with fixed quantities as expected bandwidth requirements. The work in described in [25], relaxes the last requirement, and proposes the stochastic traffic engineering framework, where the entries in the traffic matrix are not fixed values but are based on some Gaussian distribution. The authors found that the variability of the demand has a great impact on path selection. Though this last work improves the assumption on traffic matrix with fixed values, it

is still based on the ingress to egress assumption, which is known as the pipe model. The authors of [26] first proposed the hose resource provisioning model, where there is no need for a full traffic matrix but we only need to know the total traffic an border node injects/receives into/from the network. This work introduced algorithms for designing minimum cost networks based on the hose model, based on the steiner tree approach [26], while in [27] proved that the optimal hose provisioning problem is *NP*-hard and proposed some heuristics. Finally, [28] discusses the bandwidth efficiency of the hose model compared to the pipe and gives a lower bound for the hose model realisation.

Works like [12], [29] and [30] try to achieve optimal routing behaviour by appropriately configuring the shortest path routing metrics, assuming no MPLS is supported by the network. Wang et al. in [29] proved theoretically that any routing configuration, including the optimal one, could be achieved by the appropriate setting of the shortest path routing metrics.

Finally, online algorithms are mainly based on extensions of the QoS-routing [31], [32]. These approaches are heuristics, recently known in the IETF as Constraint Shortest Path First (CSPF), which utilise information kept in traffic engineering databases populated through information obtained from the routing flooding mechanisms [9] about link capacities, unreserved capacity, colour affinities etc. Other online traffic engineering approaches [33], [34] and [35] mainly focus on load balancing on multiple equal or non-equal cost paths.

3 Two-Level Intra-domain Traffic Engineering

In [36] we proposed a two-level traffic engineering system operating both at long-to-medium and medium-to-short time scales. The relevant architecture is depicted in Fig. 1.

At the long-to-medium time scale, Network Dimensioning maps the traffic requirements to the physical network resources and provides dimensioning directives in order to accommodate the predicted traffic demand. At the medium-to-short time scales, we manage the routing processes in the network, performing dynamic load balancing over multiple edge-to-edge paths, and we ensure that link capacity is appropriately distributed among the PHBs in each link by appropriately selecting the scheduling discipline and buffer management parameters. This part is realised by the Dynamic Route and Dynamic Resource Management.

Dynamic Route Management operates at the edge nodes and is responsible for managing the routing processes in the network, performing mainly dynamic load balancing. An instance of Dynamic Resource Management operates at each router and aims to ensure that link capacity is appropriately distributed among the PHBs in that link by setting the relevant buffer and scheduling parameters. By setting appropriately how the link capacity is partitioned between the several queues and if they can make use of any *unused* capacity or if they act in isolation, as in hierarchical scheduling disciplines, we can achieve the performance targets that were set by Network Dimensioning.

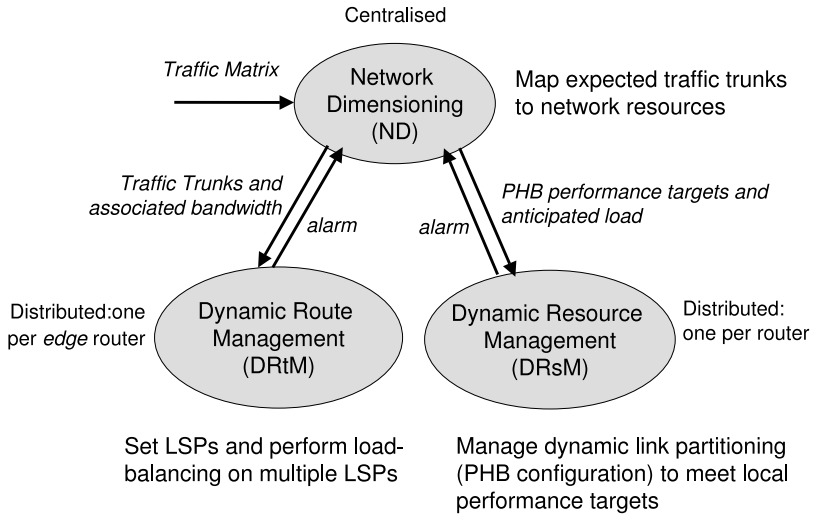


Fig. 1. Two-level Traffic Engineering

It should be noted that in this traffic engineering approach, MPLS LSPs are used purely to denote a set of explicit paths, without having explicitly assigned bandwidth within the network, while the proposals in the IETF [7], [37] and research efforts [22], [23] assume that bandwidth is assigned to LSPs. Network dimensioning comes up with a set of LSPs realising a traffic trunk and *logically* associated bandwidth, e.g. for the A-Z 1.2Mbps trunk, LSPs A-B-D-Z 0.4Mbps, A-E-Z 0.5Mbps and A-F-G-Z 0.3Mbps may be produced, but the bandwidth association is only kept at the ingress node (A in this case), with Dynamic Route Management performing load balancing of incoming traffic to those LSPs.

In summary, through this approach network provisioning is effectively achieved by taking into account both the long-term service level subscriptions in a time-dependent manner and the dynamic network conditions that are state-dependent. We argue that both levels are necessary when considering an effective traffic engineering solution, since when used independently they have shortfalls (*centralised* vs. *distributed, time-* vs. *state-dependent, static* vs. *dynamic*), which can only be overcome when they are used in conjunction.

4 Service-Driven Class-Based Traffic Engineering

The Traffic Engineering system presented in the Sect. 3 does not act in isolation but is driven by service level functions [14], not depicted in Fig. 1, for offering and establishing Service Level Agreements (SLAs). The service-driven TE work is based on the technical part of an SLA, which is called the Service Level Specification (SLS) [38]. The SLSes are used for handling the admission of service requests. These service level functions set the traffic-related target objectives of

the Traffic Engineering functions to achieve. More specifically, the service layer provides the Traffic Matrix to the Traffic Engineering functions, which specifies anticipated QoS traffic demand between network edges.

The traffic engineering is also class-based and we use the notion of quality of service class (QoS-class), which represents traffic that belongs to a particular PHB and has delay and/or packet loss requirements within a particular range (see Table 1). The entries in the Traffic Matrix are the traffic trunks. A traffic trunk represents aggregate traffic that belongs to a particular QoS-class and has a certain topological scope (see Table 1). In fact, traffic trunks are aggregates of QoS traffic having the transfer characteristics of the associated QoS-class between network edges of the provider's domain.

Traffic demand is forecasted from the current SLS subscriptions, historical data and the Service Providers' expectations (e.g. sales targets). Based on these traffic forecasts, the network is appropriately dimensioned (i.e. off-line traffic engineered) by the TE functions, in terms of PHBs and their configuration parameters and in terms of QoS route constraints.

Table 1. Definition of QoS-class and Traffic Trunk

QoS-class	: PHB,	Delay range,	Packet Loss range
Traffic Trunk : QoS-class, Ingress IP address, Egress IP address, Bandwidth			

4.1 Traffic Forecast

Traffic forecasting for Traffic Matrix derivation has attracted the attention of many researchers in the last years [39], [40], [41], with considerable results. All the relevant works up to now are based on statistical processing of measurement and historical data. We believe that in the near future advanced services will be offered and customers are going to subscribe to such services through the notion of the SLS [38]. Thus we propose that Traffic Forecasting should also take into account the customer subscriptions, in addition to network measurements. In this section we will discuss a few issues on Traffic Matrix derivation based on customer service subscriptions.

The role of the Traffic Forecast is to estimate the anticipated traffic for each forecasting period. The forecasting periods are determined based on the forecasting period schedule. For each forecasting period, four successive functions are performed: translation, mapping, aggregation, and forecasting.

Translation. The Customer SLSes stored in the repository use a customer-oriented terminology. These SLSes have the following semantics. They are the either bi-directional or unidirectional and follow either the pipe or the VPN (Virtual Private Network) model. The first function of translation is to decompose these SLSes into a number of simple unidirectional SLSes, which follow

only the pipe model (one ingress to one egress). The ingress and egress are specified in geographical terms, therefore there is a need to translate those into IP source-destination addresses and from them to infer the specific ingress-egress addresses of our Autonomous System (AS). Services are given names (Olympic Gold/Silver/Bronze Service, Virtual Leased Line). This terminology must use networking semantics (throughput, delay, loss). So, another important aspect of translation is from the customer SLS to the appropriate network parameters.

Mapping. The simple unidirectional SLSes could potentially request (or be translated to) any value of delay, loss or throughput, while the network supports only a few discrete values. The mapping function is therefore to map the SLS requirements to the services actually supported by the network, i.e. the QoS classes (see Table 1). Another important function is to map the IP source-destination addresses. To illustrate the mapping function, consider the following simple example. A customer SLS may require "1Mbps of Virtual Leased Line Premium service with 50ms delay between London and Manchester, ...". The translation function would translate this Customer SLS into to two simple unidirectional SLSes: "1Mbps, EF, 50ms delay, from 122.12.2.143, egress 103.124.32.111, ..." and another one with the same values but in the opposite direction. Suppose the network offers 2 services: a 10ms and 100ms delay premium virtual wire services, therefore we need to map the SLS to the 10ms service. The mapping is static as far as the mapping algorithms are fixed and deterministic. The result is a list of SLSes that are active during the next provisioning period and are defined in network terms.

Aggregation. Up to this point of the forecasting procedures we had information proportional to the number of customers and SLSes. For scalability, we base our TE solution on traffic loads per ingress/egress pair and per class of service. The simple unidirectional SLSes are aggregated into traffic trunks by "adding" their throughput requirements if and only if they have the same ingress/egress pair and they require a similar treatment, i.e. they belong to the same QoS class.

Forecasting has two main functions. On one hand, an over-subscription factor per QoS class is included. This factor is defined as the ratio of the capacity reserved by all the SLSes in a given QoS class to the capacity expected to be actually used. For expensive SLS types, the over-subscription factor is likely to be one. For cheaper services, the factor may be larger. On the other hand, at this stage we have all the ingress-egress demand (traffic matrix). It is possible to run an extrapolation algorithm utilising the information on the history of traffic matrices. Candidate algorithms are the spline or more generally any polynomial extrapolation method.

4.2 Provisioning and Forecasting Scheduling

In this section we describe the timing and scheduling properties of our system. We assume that scheduling in the proposed model is performed at two levels:

1. The Network Provisioning Scheduler, which defines provisioning periods.
2. The Traffic Forecast Scheduler, which defines forecasting periods.

Network Provisioning Scheduler. The proposed Traffic Engineering System aims to provision the network in order to provide the required QoS to contracted SLSes while at the same time optimising the usage of resources. As the traffic requirements that are derived from the SLSes change over time, the provisioning guidelines need to be updated. This is performed periodically through the Network Provisioning Scheduler. The system is required to recalculate a set of provisioning guidelines for each provisioning period. As the current period comes to an end, the Provisioning Scheduler will trigger re-provisioning of the network. Typical frequencies for the scheduler could be once a day or once a week.

Traffic Forecast Scheduler. As the provisioning periods are quite long (days, weeks), the traffic requirements may vary during a provisioning period. The granularity of the Network Provisioning Scheduler is not fine enough to optimise the configuration of the network. We therefore use a second scheduler, the Traffic Forecast Scheduler, which has the following characteristics (see also Fig. 2):

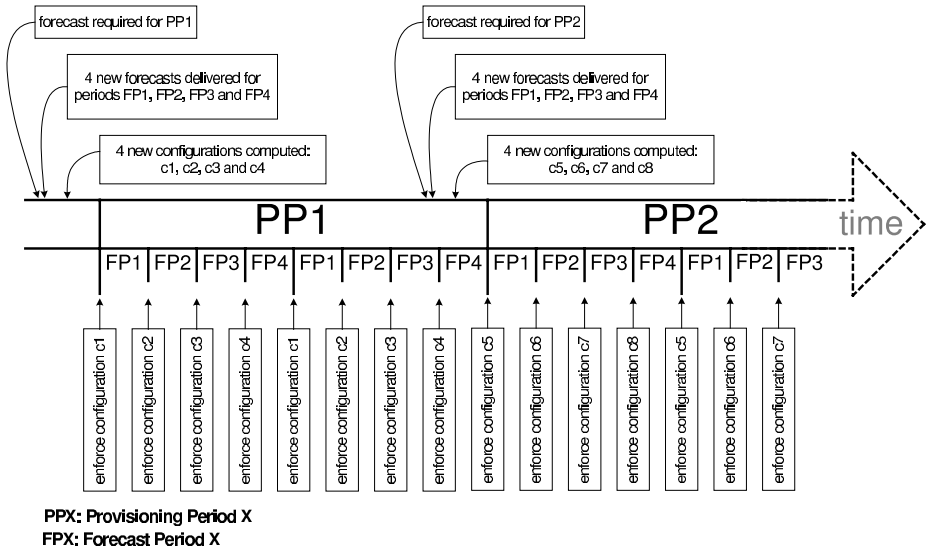


Fig. 2. Two-level scheduling

- It is not necessarily periodic (could schedule one 4 hour period followed by one 10 hour period)
- For all forecasting periods $FP(i)$ that constitute a provisioning period PP, it holds: $FP(i) = PP$

The principle behind the double scheduling is that network provisioning is invoked once every provisioning period, and calculates multiple configurations, which will be enforced at each forecasting period triggered by the Traffic forecast scheduler.

Scheduling Example. To clarify the roles of each scheduler, this section provides some examples. Consider that provisioning period is a week and that Traffic Forecast Scheduler defines 4 6-hour forecasting periods (forecasting period being periodic in this case). This means that once a week, 4 logical topologies are calculated: one for the morning, one for the afternoon, one for the evening and one for the night (6 hour periods). During each day of the week, Traffic Forecast Scheduler triggers at 06:00 to enforce the morning configuration, at 12:00 to enforce the afternoon configuration, at 18:00 to enforce the evening configuration and at 24:00 to enforce the night configuration. This is shown in Fig. 2.

There is no requirement for the Traffic Forecast Scheduler to be periodic. For example, we could have 4 forecasting periods per day, morning, afternoon, evening and night with different lengths as shown in Table 2.

Table 2. Forecasting periods schedule example

Morning	: 08:00 - 12:00
Afternoon	: 12:00 - 17:00
Evening	: 17:00 - 22:00
Night	: 22:00 - 08:00

Another more realistic and more complex example could be to have 5 forecasting periods: weekday morning, afternoon, evening, night, and all weekend. The interesting feature of this arrangement is that the Traffic Forecast Scheduler does not trigger the enforcement of the configurations in a simple cyclical fashion: the 4 weekday configurations must be implemented cyclically for 5 days, and the weekend configuration is enforced for the remaining two days of the week.

5 Network Dimensioning

Network Dimensioning (ND) is time- and state-dependent offline traffic engineering. It performs automated provisioning and is responsible for the long to medium term configuration of the network resources. By configuration we mean the definition of LSPs as well as the anticipated loading for each PHB on all interfaces, which are subsequently being translated by Dynamic Resource Management into the appropriate scheduling parameters (e.g. priority, weight, rate limits) of the underlying PHB implementation. The values provided by Network Dimensioning are not absolute but are in the form of a range, constituting directives for the function of the PHBs, while for LSPs they are in the form of multiple paths to enable multi-path load balancing. The exact PHB configuration values and the load distribution on the multiple paths are determined by Dynamic Resource Management and Dynamic Route Management respectively,

based on the state of the network, but should always adhere to the Network Dimensioning directives.

Network Dimensioning runs periodically, getting the expected traffic per PHB in order to be able to compute the provisioning directives. The objectives are both traffic and resource-oriented. The former relate to the obligation towards customers, through the SLSes. These obligations induce a number of restrictions about the treatment of traffic. The resource-oriented objectives are related to the network operation, more specifically they are results of the high-level business policy that dictates the network should be used in an optimal fashion. The basic Network Dimensioning functionality is summarised in Fig. 3.

Input: Network topology, link properties (capacity, propagation delay, PHBs)
Pre-processing: Request traffic forecast, i.e. the potential traffic trunks Obtain statistics for the performance of each PHB at each link Determine the maximum allowable hop count per traffic trunk according to the PHB statistics
Optimisation phase: Start with an initial allocation (e.g. using the shortest path for each traffic trunk) Iteratively improve the solution such that for each traffic trunk find a set of paths: -The minimum bandwidth requirements of the traffic trunk are met -The hop-count constraints K is met (delay/ loss requirements are met) -The overall cost function is minimised
Post-processing: Allocate any extra capacity to the resulted paths of each PHB according to resource allocation policies Sum the path requirements per link per PHB, give minimum (optimisation phase) and maximum (post-processing phase) allocation directives to Dynamic Resource Management Give the appropriate multiple paths calculated in the optimisation phase to Dynamic Route Management

Fig. 3. Network Dimensioning functionality

Congestion is the main cause of network performance degradation that influences both these traffic engineering objectives. Two are the main reasons behind congestion: either the demand exceeds the network capacity, or the traffic is not spread optimally, causing parts of the network to be over-utilised while others are under-utilised. The first reason can only be handled by adequate network planning, i.e. adding more physical resources. The latter can be handled by adequate

bandwidth and routing management, and this is what Network Dimensioning is aiming for.

Network Dimensioning has as input the network topology (including link capacities), the estimated traffic matrix (expected traffic trunks, see Sect. 4.1), and resource-oriented policy directives. It will provide as an output a list of explicitly routed paths for each source destination included in the traffic matrix, and the capacity requirements for each PHB, and therefore physical queue for every interface of all the network nodes.

5.1 Network and Traffic Model

The network is modeled as a capacitated directed graph $G = (V, E)$, where V is the set of nodes and E the set of links. Each link $l \in E$ is specified by the pair $l = (v_{l,i}, v_{l,e}$ where $v_{l,i}$, $v_{l,e}$ are the nodes $\in V$, where traffic is entering (ingress) and exiting the link (egress) respectively.

With each unidirectional link l we associate the following parameters: the link physical capacity c_l, the link propagation delay d_l^{prop}, the set of PHBs H, supported by the link. For each PHB $h \in H$ we associate a bound d_l^h (deterministic or probabilistic depending on the PHB) on the maximum delay incurred by aggregate traffic entering link l and belonging to h. and a bound on the loss probability p_l^h of the aggregate traffic entering link l and belonging to h.

The basic traffic model of network provisioning is the traffic trunk (see Sect. 4). The set of all traffic trunks is denoted by T. Each trunk $t \in T$ is associated with a bandwidth requirement, B_t. The following information about each traffic trunk is available because of the QoS class definition (see Table 1):

- The PHB $h \in H$ of the traffic carried on the trunk.
- The bound D_t (deterministic or probabilistic depending on the PHB) on maximum end-to-end delay expected between the ingress and egress nodes. This might be the minimum value of the delay range of the QoS-class definition.
- The maximum end-to-end loss probability, P_t required between the ingress and egress nodes. Similarly to the previous case this might be the minimum value of the loss range of the QoS class definition.
- The bandwidth B_t requirement.

5.2 Cost Definition and Optimisation Objectives

We need to provide a set of routes for each traffic trunk. For each ingress-egress pair, these routes are implemented as LSPs at the routers. We also need to provide the amount of bandwidth each route is expected to carry, and the amount of bandwidth that is to be allocated to the PHBs at the various interfaces in the network.

The *primary objective* of such an allocation is to ensure that the requirements of each traffic trunk are met as long as the traffic carried by each trunk is at its specified minimum bandwidth. This objective ensures that our SLS requirements

are met. The objective is to provide a feasible solution (i.e., routes and route bandwidths respecting the link capacities) that satisfies the SLSes delay and loss constraints.

However, with the possible exception of heavily loaded conditions, there will generally be multiple feasible solutions. Hence, the design objectives can be further refined to incorporate other requirements such as:

(a) avoid overloading parts of the network while other parts are underloaded. This way, spare bandwidth is available at various parts of the network to accommodate unpredictable traffic requests. In addition, in case of link failures, smaller amounts of traffic will be disrupted and will need to be rerouted, and
(b) provide overall low network cost (load).

The last two requirements do not lead to the same optimisation objective. In any case, in order to make the last two requirements more concrete, the notion of *load* has to be quantified. Various definitions are possible. In general, the load (or cost) on a given link is an increasing function of the amount of traffic the link carries. This function may refer to link utilisation or may express an average delay, or loss probability on the link. In the context of this work, the QoS seen by the traffic using the different PHBs varies, so the link load induced by the traffic of each PHB may vary. This leads us to the following general form of the cost of link l.

Let x_l^h denote the capacity demand (flow) for PHB $h \in H$ satisfied by link l. The link cost induced by the load on PHB h is a convex function, $f_l^h(x_l^h)$, increasing in x_l^h. The total link cost per link is defined as:

$$F_l(\bar{x}_l) = \sum_{h \in H} f_l^h(x_l^h) \tag{1}$$

where $\bar{x}_l$ is the vector of demands for all PHBs of link l.

Provided that appropriate buffers have been provided at each router and the scheduling policy has been defined, then $f_l^h(x_l^h)$ may specify the *equivalent capacity* needed by PHB h on link l in order to satisfy the loss probability associated with that PHB. Hence, the total cost per link is the total equivalent capacity allocated on link l. Note that with this approach the link costs are very naturally defined. The drawbacks are: 1) The cost definition depends on the PHB implementation at the routers, 2) The cost functions may not be known, or may be too complex. Hence approximate cost functions must be used. An example may be linear function of x_l^h, $f_l^h(x_l^h) = a_l^h x_l^h$, where a_l^h is a constant. Other examples may be found in [42].

We express objectives (a) and (b) formally as follows:
(a) avoid overloading parts of the network, i.e.

$$\text{minimise } \max_{l \in E} F_l(\bar{x}_l) \tag{2}$$

(b) minimize overall network cost, i.e.

$$\text{minimise } \sum_{l \in E} F_l(\bar{x}_l) \tag{3}$$

We have proposed [42] a new objective function that compromises between the previous two. More specifically,

$$\text{minimise} \sum_{l \in E} [F_l(\bar{x}_l)]^n \,, \quad n \in [1, \infty) \tag{4}$$

When $n = 1$ the objective defined in (4) reduces to the one in (3). As n increases the objective defined by (4) increasingly favours solutions which adhere to the objective defined by (2). When $n \to \infty$, it can be shown that (4) reduces to (2), since the factor with the maximum F_l will grow much faster than the others, and thus it will be increasingly the most important factor of the sum in (4).

Equation (4) uses a non-linear combination of the cost functions (1), thus the resulting optimisation problem is of non-linear nature even if $f_l^h(x_l^h)$ is a linear cost function for each link for each PHB. The resulting optimisation problem is a network flow [20] problem and considering the non-linearity of the objective function (4), the optimal solution can be based on the general gradient projection method [43]. This is an iterative method, where we start from an initial feasible solution, and at each step we find the minimum first derivative of the cost function path and we shift part of the flow from the other paths to the new path, so that we improve our objective function. The details of how we use this procedure can be found in [42].

Delay and Loss Constraints. Each traffic trunk is associated with an end-to-end delay and loss probability, constraint of the traffic belonging to the trunk. Hence, the trunk routes must be designed so that these two QoS constraints are satisfied. Delay and loss constraints are both on additive path costs under specific link costs. However, the problem of finding routes satisfying these constraints is in general NP-complete [31], [44]. Given that this is only part of the problem we need to address, the problem in its generality is rather complex.

Fortunately, we can make a reasonable simplification. Usually the measured loss probabilities and delay for the same PHB on different nodes (routers) are of similar order. Therefore we use the maximum delay, and maximum loss probability of a particular PHB over the whole network, in order to translate the delay and loss constraints to an upper bound on the number of hops along a route. During network operation, the average delay and loss at each link/PHB are monitored and the maximum respective values are used in the next provisioning period. This is relatively conservative and assuming the network is not overloaded, the delay and loss constraints of the traffic trunks should be met.

Note that under normal operating conditions, the delay and loss at each link/PHB for "premium trunks" should be low, assuming correct network dimensioning and admission control functionality at the ingress nodes. Problems with dimensioning or admission control may cause increased delay and loss, and this will be realised by network monitoring and subsequently dimensioning, triggering in the short term different (stricter) operating policies and in the longer term different network planning with more physical resources.

5.3 Simulation Results

In the simulation results we present in this section we set the initial feasible solution (step 0) of the iterative procedure of the solution [42] to be the same as if the traffic trunks were to be routed with a shortest path first (SPF) algorithm. That corresponds to the case that all the traffic of a particular class from an ingress to an egress will be routed through the same shortest path according to some weight (routing metric). If there exist more than one such shortest paths, the load is distributed equally among them. The metric we are using for the SPF algorithm is set to be inversely proportional to the physical link capacity. The scenario we are using for step 0 described above is the norm in today's operational networks. The experiments shown in this section correspond to only one forecasting period, i.e. one run of the provisioning algorithm.

We define as total throughput of a network the sum of the capacities of the first-hop links of all the edge nodes (see Fig. 4). This is actually an upper bound of the throughput and in reality it is a much greater than the real total throughput a network can handle. In our experiments we used 70% load of the total throughput of the network, as the highly loaded condition, and a 40% load as a medium loaded one.

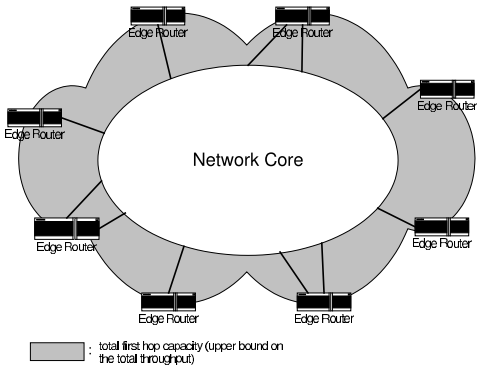


Fig. 4. Illustration of the total throughput as the sum of first hop links

Figure 5 shows the link loads for random 34-link topologies [45], at the first step and after the algorithm has run. It is clear that at step 0 solution (dark bars), which corresponds to the SPF with equal cost multi-path distribution enabled, parts of the network are over utilised while some links have no traffic at all. The final step (pale dry bars), which corresponds to the final output of our provisioning algorithm, balances the traffic over the whole network. Note that the largest over utilised link at step 0 is not necessarily the largest at the final solution.

Figure 6 shows the mean and standard deviation of the link loads after each iteration of the algorithm. As we can see the load becomes more balanced over the network after each iteration (standard deviation reduces). We run those

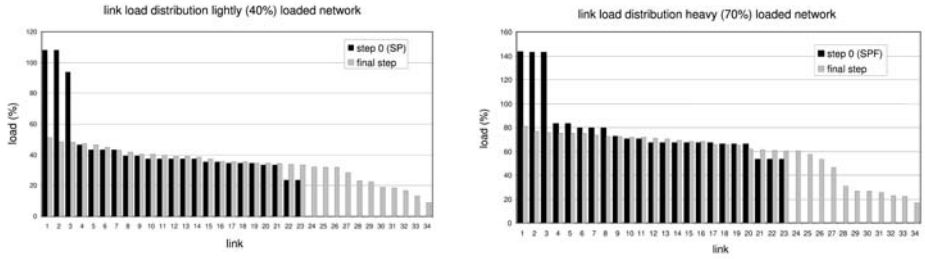


Fig. 5. Link load distribution of after the first and the last step

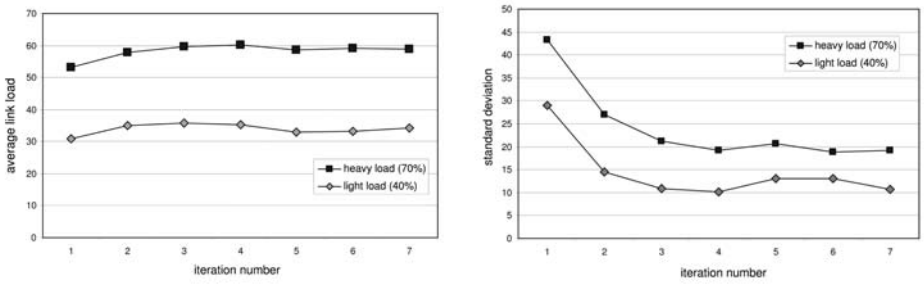


Fig. 6. Average and standard deviation of link load per iteration

experiments with the exponent n, see (4), equal to 2. This value compromises between minimising the total (sum) of link costs and minimising the maximum link load. These two objectives generally lead to different solutions, with the first favouring path solutions with the least number of links while the other does not care about the number of links but only for the maximum link load (and therefore the deviation from the mean). We can observe the effect of these different objectives at the various ups and downs over the various steps of the algorithm.

The same behaviour observed with the network is also observed with larger random and transit-stub topologies [45]. We have experimented with large topologies up to 300 nodes under various loading scenarios and we observed the same behaviour as above. For the the details on more complex scenarios with large topologies and results on QoS constraints we refer the reader to reference [42].

Finally, as far as the the average running times of the various experiments conducted are concerned, we observed that even for quite large networks the running times were acceptable. For example for 300-node networks with more than 2000 links, for medium load the running time was on average about 17 minutes, and for high load about 25 minutes on a standard PC with 1GHz processor. These times are perfectly acceptable taking into account the timescale of the provisioning system operation (see Sect. 4.2).

6 Policy Extensions to Intra-domain Traffic Engineering

Policy-based Management has been the subject of extensive research over the last decade. Policies are seen as a way to guide the behaviour of a network or distributed system through high-level, declarative directives. The IETF has been investigating policies as a means for managing IP-based multi-service networks, focusing more on the specification of protocols (e.g. COPS) and the object-oriented information models for representing policies. We view policy-based management as a means of extending the functionality of management systems dynamically, in conjunction with pre-existing *hard-wired* logic [4]. Policies are defined in a high-level declarative manner and are mapped to low-level system parameters and functions, while the system intelligence can be dynamically modified added and removed by manipulating policies. Inconsistencies in policy-based systems are quite likely since management logic is dynamically being added, changed or removed without the rigid analysis, design, implementation, testing and deployment cycle of "hard-wired" long-term logic. Conflict detection and resolution is required in order to avoid or recover from such inconsistencies.

In the architecture shown in Fig. 1, Network Dimensioning besides providing long- term guidelines for sharing the network resources, it can also be policy influenced so that its behaviour can be modified dynamically at run-time reflecting high-level, business objectives. The critical issue for designing a policy-enabled resource management system is to specify the parameters influenced by the enforcement of a policy that will result in different allocation of resources in terms of business decisions. These policies that are in fact management logic, are not hard-wired in the system but are downloaded on the fly while the system is operating.

We extended [36], [46] the traffic engineering system shown in Fig. 1 to be able to drive its behaviour through policies. The resulting extended system architecture is depicted in Fig. 7. The Policy extensions include components such as the Policy Management Tool, Policy Repository, and the Policy Consumers.

A single Policy Management Tool exists for providing a policy creation environment to the administrator where policies are defined in a high-level declarative language and after validation and static conflict detection tests, they are translated into object-oriented representation (information objects) and stored in a repository. The Policy Repository is a logically centralised component but may be physically distributed since the technology for implementing this component is the LDAP (Lightweight Directory Access Protocol) Directory. After the policies are stored, activation information may be passed to the responsible Policy Consumer in order to retrieve and enforce them. The Policy Consumer can be seen as a co-located Policy Decision Point (PDP) and Policy Enforcement Point (PEP) with regards to the IETF Policy Framework [47].

In Fig. 7 the representation of the policies at every level of the framework is also depicted, showing that every policy is going through two stages of translation/refinement in its life-cycle in order to be enforced. Starting from the high-level specification to the object-oriented format (LDAP objects) and then from the LDAP objects to a script that is interpreted on the fly, complement-

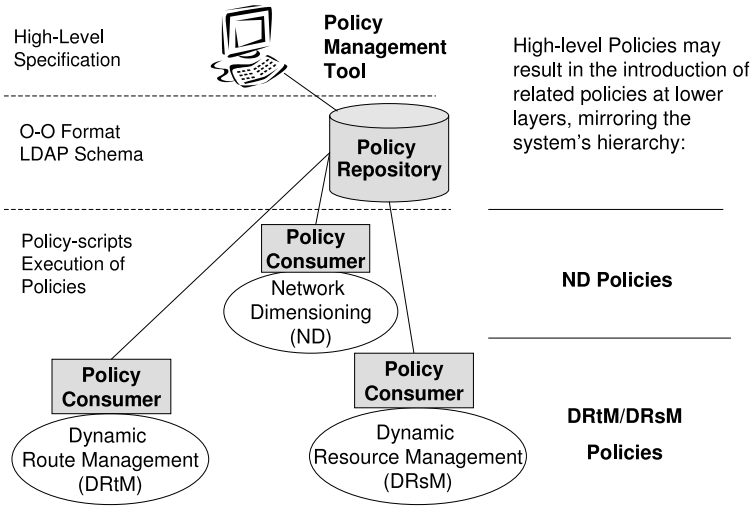


Fig. 7. Policy-driven Traffic Engineering Architecture

ing this way conceptually the management intelligence of the above layer in the hierarchy.

For example, a policy enforced on the Dynamic Resource Management is actually enhanced management logic that conceptually belongs to the Network Dimensioning layer of our system. Although policies may be introduced at every layer of the system, higher-level policies may possibly result in the introduction of related policies at lower levels, forming a policy hierarchy mirroring the management system's hierarchy. This means that a policy applied to a hierarchical system might pass through another stage of translation and refinement that will generate the policies that are enforced in the lower levels of the system. It is questionable if the automation of this process is feasible without human intervention. A more detailed discussion on policy-based hierarchical management systems can be found in [48].

In the rest we will focus on the policies we identified for the specific domain defined by the functionality of Network Dimensioning as described in see Sect. 5.2.

6.1 Network Dimensioning Policies

We identify two categories of policies, *initialisation* and *resource provisioning* policies. The initialisation are policies that result in configuring with initial values the variables, which are essential for the functionality of Network Dimensioning and do not depend on any state but just reflect decisions of the policy administrator. The second category, the resource provisioning policies, are the policies that influence the way Network Dimensioning calculates the capacity

allocation and the path creation configuration of the network. Such policies are those that their execution is based on the input from the traffic forecast module and on the resulting configuration of the network.

Since dimensioning is triggered mainly periodically, the policy administrator should specify this period. The priority of this policy should be specified in order not to cause any inconsistencies when re-dimensioning is triggered by notifications sent from the dynamic control parts of the system, that is when Dynamic Route Management and Dynamic Resource Management are unable to perform an adaptation of the network with the current configuration.

Another parameter that should be initialised by the policy system is the cost-function $f_l^h(x_l^h)$ used by the Network Dimensioning algorithm (see Sect. 5.2). The administrator should be able either to choose between a number of pre-specified cost functions and/or setting values to parameters in the desired function. For example, the approximate cost function used by Network Dimensioning could be linear function of the bandwidth allocated to a PHB, e.g. $f_l^h(x_l^h) = a_l^h x_l^h$, where a_l^h is a constant and x_l^h is the bandwidth allocated to PHB $h \in H$ on link l. In this case the initial value given to a_l^h could be configurable, and thus it could be specified by the policy administrator. Depending on the importance of a particular link l and of a particular PHB h, the constant can be increased and thus increasing the cost of using that PHB on this link.

Another constraint parameter which can be configurable by policies is the maximum number of alternative paths that Network Dimensioning defines for every traffic trunk for the purpose of load balancing. Finally, the exponent n of the objective function, as defined in (4), is another parameter that is specified by policies allowing the administrator to choose the relative merit between the two optimisation objectives, i.e. achieve low overall cost and avoid overloading parts only of the network.

Resource provisioning policies are more advanced compared to initialisation policies since their enforcement is not a simple parameter setting but requires the invocation of some logic. The policy administrator should be able to specify the amount of network resources (giving a minimum, maximum or a range) that should be allocated to each PHB. This will cause Network Dimensioning to take into account this policy when calculating the new configuration for this PHB. More specifically, Network Dimensioning should allocate resources in a way that does not violate the policy and then calculate the configuration taking into account the remaining resources.

A more flexible option is for the policy administrator to indicate how the resources should be shared in specific (critical) links. After the optimisation phase ends, Network Dimensioning enters a post-processing stage (see Fig. 3) where it will try to assign the residual physical capacity to the various PHBs. The way this distribution of spare capacity is done is left to be defined by policies that indicate whether it should be done proportionally to the way resources are already allocated or in a max-min fair way or it can be defined explicitly for every PHB. Another related policy is to specify the way the capacity allocated

to each PHB should be reduced because the link capacity is not enough to satisfy the predicted requirements as given in the Traffic Matrix.

Network Dimensioning translates the delay and loss requirements on an upper bound on the number of hops per route, the way this translation is done can also be influenced by policy rules. For example, the safest approach to satisfy the traffic trunk requirements would be to assume that every link and node belonging to the route induces a delay equal to the maximum delay induced by an interface along the route. So, this policy rule will allow the administrator to decide if the maximum, average or some percentile of the actual measured delay or loss induced by an interface along the route should be used to derive the hop count constraint.

Policies that allow the administrator for a particular reason to explicitly specify an LSP that a traffic trunk should follow can also be defined. Of course, the resources for the administratively set LSP should be excluded from the available resources before the Dimensioning algorithm starts.

6.2 Policy Enforcement Example

The policy rule example concerns the effect of the cost function exponent n, see (4) in Sect. 5.2, on the capacity allocation of the network. As we discussed in Sect. 5.2 when we increase the cost function exponent n the optimisation objective that avoids overloading parts of the network is favoured compared to achieving overall low cost.

If it is required to keep the load of every link below a certain point then the administrator will have enter the appropriate policy in the Policy Management Tool using our proprietary policy language, which is then translated in LDAP objects according to an LDAP schema based on the Policy Core LDAP Schema [49] and stored in the Policy Repository. The syntax of our language as well as the extension to the Policy Core Information Model [50] with specific classes that reflect the policies described in the previous section are presented in [46]. The policy is entered with the following syntax:

```
if maxLinkLoad > 80% then decrease maxLinkLoad          (1)
```

After this rule is correctly translated and stored in the repository, the Policy Management Tool notifies the Policy Consumer which is associated with Network Dimensioning, that a new policy rule is added in the repository. The Policy Consumer then retrieves all the associated objects with this policy rule. From these objects the consumer generates code that is interpreted and executed representing the logic added in our system by the new policy rule.

The pseudo-code produced by the Policy Consumer for policy (1) is shown in Fig. 8. The produced code will start by setting the exponent n to 1 and then will run the iterative optimisation algorithm, see `Optimisation phase` in Fig. 3. Then it will check if the value of `maxLinkLoad`, which represents the utilisation of the link with the maximum load as resulted from the optimisation algorithm, is less than the required value defined by the policy rule. If `maxLinkLoad` is above

the aforementioned value, it will increase n an it will run the optimisation algorithm again until the policy is enforced. Note that as n increases the algorithm favours longer paths and thus we may not reach the desired solution as far as the hop-count constraint is concerned, but in any case if there is a feasible solution for enforcing the policy it will be found.

```
maxLinkLoad: maximum link load
 utilisation after optimisation
n=1: cost function exponent

run_optimisation_algorithm  n
while ( maxLinkLoad > 80 )
  n = n+1
  run_optimisation_algorithm n
```

Fig. 8. Pseudo-code produced when enforcing policy (1)

Figure 9 plots the maximum link load utilisation against the exponent of the objective function. The policy objective is achieved when $n = 4$, thus the enforcement of the policy rule caused the optimisation algorithm to run for 4 times until the maximum link load utilisation at the final step drops below 80%.

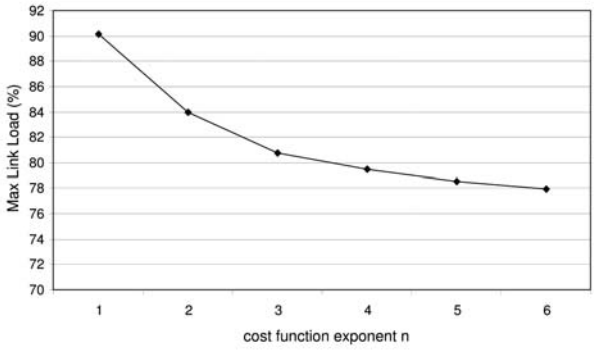


Fig. 9. Effect of the cost function exponent on the maximum link load utilisation

7 Interdomain Routing with BGP

Internet routing is handled by two distinct protocols with different objectives. Inside a single domain, link-state intradomain protocols such as OSPF or IS-IS distribute the entire network topology to all routers and select the shortest path

according to a metric chosen by the network administrator. Across interdomain boundaries, the interdomain routing protocol is used to distribute reachability information and to select the best route to each destination according to the policies specified by each domain administrator. For scalability and business reasons, the interdomain routing protocol is only aware of the interconnections between distinct domains, it does not know any information about the structure of each domain.

7.1 BGP Basics

The current de facto standard interdomain routing protocol is the Border Gateway Protocol (BGP) [51, 52]. In the BGP terminology, domains are called Autonomous Systems (AS) since these are usually managed by different independent companies. BGP is a *path-vector protocol* that works by sending *route advertisements*. A route advertisement indicates the reachability of a network which is a set of contiguous IP addresses represented by a *network address* and a *network mask* called a prefix. For instance, `192.168.0.0/24` represents a block of 256 addresses between `192.168.0.0` and `192.168.0.255`. A BGP router will advertise a route to a network because this network belongs to the same AS or because a route advertisement for this network was received from another AS. If a router of `ASx` sends a route advertisement for network N to a router of `ASy`, this implies that `ASx` accepts to forward IP packets with destination N on behalf of `ASy`.

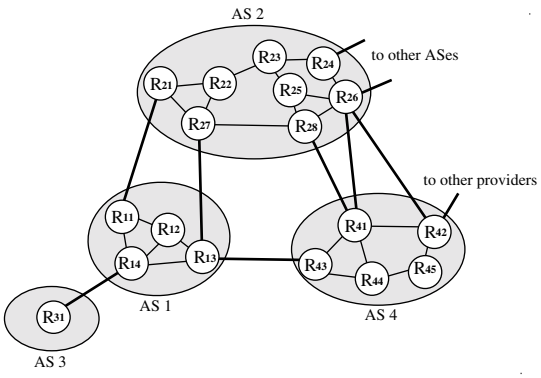


Fig. 10. A simple Internet

A route advertisement is mainly composed of the prefix of the network and the `next-hop` which is the IP address of the router that must be used to reach this network. A route advertisement also contains the `AS-path` attribute which contains the list of all the transit AS that must be used to reach the announced

network. The `AS-path` has two important functions in BGP. First, it is used to detect routing loops. A BGP router will ignore a received route advertisement with an `AS-path` that already contains its AS number. Second, the length of the `AS-path` can be considered as a kind of route metric. A route with a shorter `AS-path` will usually be considered better than a route with a longer one.

Besides the `AS-Path`, a route advertisement may also contain several optional attributes such as `local-pref`, `multi-exit-discriminator` (`med`) or `communities` [51, 52].

7.2 Route Filtering

Inside a single domain, all routers are considered as "equal" and the intradomain routing protocol announces all known paths to all routers. In contrast, in the global Internet, all ASes do not play the same role and an AS will seldom agree to provide a transit service for all its neighbour ASes toward all destinations. Therefore, BGP allows a router to be selective in the route advertisements that it sends to neighbour eBGP routers. To better understand the operation of BGP, it is useful to consider a simplified view of a BGP router as shown in Figure 11.

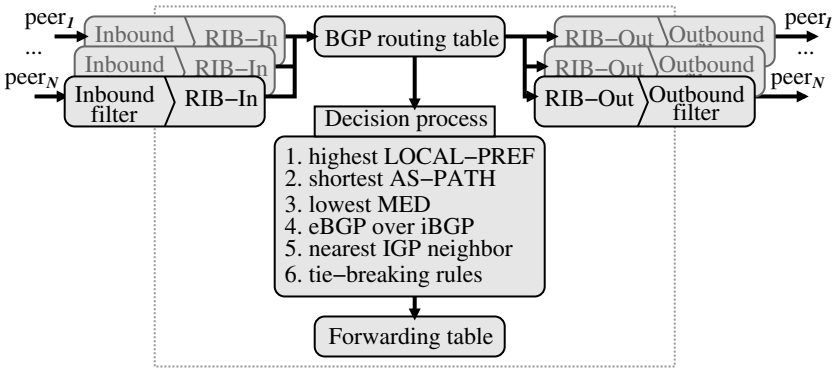


Fig. 11. Simplified operation of a BGP router.

A BGP router processes and generates route advertisements as follows. First, the administrator specifies, for each BGP peer, an input filter (Figure 11, left) that is used to select the acceptable advertisements. For example, a BGP router could only select the advertisements with an `AS-Path` containing a set of trusted ASes. Once a route advertisement has been accepted by the input filter, it is placed in the BGP routing table, possibly after having updated some of its attributes. The BGP routing table thus contains all the acceptable routes received from the BGP neighbours.

Second, on the basis of the BGP routing table, the BGP decision process (Figure 11, center) will select the best route toward each known prefix. Based on the `next-hop` of this best route and on the intradomain routing table, the router will install a route toward this network inside its forwarding table. This table is then looked up for each received packet and indicates the outgoing interface which must be used to reach the packet's destination.

Third, the BGP router will use its output filters (Figure 11, right) to select among the best routes in the BGP routing table the routes that will be advertised to each BGP peer. At most one route will be advertised for each distinct reachable prefix. The BGP router will assemble and send the corresponding route advertisements after a possible update of some of their attributes.

The input and output filters used in combination with the BGP decision process are the key mechanisms that allow a network administrator to support within BGP the business relationships between two ASes. Many types of business relationships can be supported by BGP. Two of the most common relationships are the customer-to-provider and the peer-to-peer relationships [53]. With the *customer-to-provider* relationship, a customer AS pays to utilise a link connected to its provider. This relationship is the origin of most of the interdomain cost of an AS. A stub AS usually tries to maintain at least two of these links for performance and redundancy reasons [53]. In addition, larger ASes typically try to obtain *peer-to-peer* relationships with other ASes and then share the cost of the link with the other AS. Negotiating the establishment of those *peer-to-peer* relationships is often a complicated process since technical and economical factors, as stated in [54], need to be taken into account.

To understand how these two relationships are supported by BGP, consider Figure 10. If `AS3` is `AS1`'s customer, then `AS3` will configure its BGP router to announce its routes to `AS1`. `AS1` will accept these routes and announce them to its peer (`AS4`) and upstream provider (`AS2`). `AS1` will also announce to `AS3` all the routes it receives from `AS2` and `AS4`. If `AS1` and `AS4` have a peer-to-peer relationship on the link between R_{13} and R_{43}, then router R_{13} will only announce on this link the internal routes of `AS1` and the routes received from `AS1`'s customer (i.e. `AS3`). The routes received from `AS2` will be filtered and thus not announced on the $R_{13} - R_{43}$ link by router R_{13}. Due to this filtering, `AS1` will not carry traffic from `AS4` toward `AS2`.

7.3 Decision Process

A BGP router receives from each of its peers one route toward each destination network. The BGP router must then identify the best route among this set of routes by relying on a set of criteria known as the *Decision Process*. Most BGP routers apply a decision process similar in principle to the one shown in Figure 11. The set of routes with the same prefix are analysed by the criteria in the order indicated in Figure 11. These criteria act as filters and the N^{th} criterion is only evaluated if more than one route has passed the $N - 1^{th}$ criterion. It should be noted that most BGP implementations allow the network administrator to optionally disable some of the criteria of the BGP decision process.

In most BGP implementations, the set of criteria through which the router goes to select a best route toward a given destination is similar to what follows. First, the router checks that the routes received from its peers have a reachable `next-hop`, meaning that the IP routing table must contain a route toward this `next-hop`. If more than one route with a reachable next hop exists the router will then use preferences configured by the router administrator. Such preferences may be defined locally to a router with the `weight` parameter or shared over iBGP (interior BGP) sessions with the `local-pref` attribute. The router keeps routes with the highest `weight` and then routes with the highest `local-pref`. If after this criterion more than one route remain, the length of the `AS-Path` which acts as the BGP metric is used to compare routes. The length of the `AS-Path` is seen as a measure of the quality of the route and one usually expect that the route with the shortest `AS-Path` is the best.

If at this point the decision process has not yet identified the best route toward the given destination, that means that it has to select one among a set of equal quality routes. The remaining criteria were added for this purpose. The `multi-exit-discriminator` or `med` can be used to compare routes which were received from different routers of the same AS. The route with the lowest `med` is preferred. This criterion is not always enabled because the decision process can be influenced by the remote peers which set the value of the `med`. After the `med`, the decision process prefers routes learned over an eBGP session to routes learned over an iBGP session. The router gives then the preference to routes that can be reached by the closest BGP next hop. If after all these criteria, there is still more than one candidate route, tie-breaking rules are applied. Usual criteria are to keep the oldest route (this minimises route-flapping) or to prefer the route learned from the router with the lowest ID.

8 Characteristics of Interdomain Traffic

To obtain a better understanding of the characteristics of interdomain traffic, we have relied on Netflow [55] traces of two different ISPs. Netflow is a traffic monitoring facility supported by Cisco routers. When enabled, the router regularly transmits some information about all layer-4 flows (TCP connections and UDP flows) that passed through it to a close-by monitoring station. With Netflow, the monitoring station knows the starting and ending timestamps of all layer-4 flows as well as the flow volume (in bytes and packets) and the transport protocol and port numbers. Netflow is often used for billing purposes or by ISPs that need to better understand the traffic inside their network. Compared to the traditional packet-level traces that are often analysed, Netflow has the advantage of being able to monitor multiple links during long periods of time. The main drawback of Netflow is that it does not capture the very short-term variations of the traffic, but this is not a problem in our context of interdomain traffic engineering which tackles medium to long-term traffic variations.

The only characteristics common to both ISPs is that they do not offer transit service. Besides this, they serve very different customers and it can be

expected that these customers have different requirements on the network. Due to technical reasons, it was unfortunately impossible to obtain traces from the two studied ISPs covering the same period of time.

The first trace was collected in December 1999 and covers 6 successive days of all the interdomain traffic received by BELNET. BELNET is the ISP that provides connectivity for the research and education institutions located in Belgium. At that time, BELNET was composed of a 34 Mbps star-shaped backbone linking the major universities. Its interdomain connectivity was mainly provided through 34 and 45 Mbps links to the transit service from two commercial ISPs. In addition, BELNET had a 45 Mbps link to the European research network, TEN-155, and was present at the BNIX and AMS-IX interconnection points with a total of 63 peering agreements in operation. Although some universities provided dialup access for their students, the typical BELNET user had a 10 Mbps access link to the BELNET network through their university LAN. During the 6 days period, BELNET received 2.1 terabytes of data. BELNET is representative of research networks and could also be representative of an ISP providing services to high bandwidth users with cable modem or ADSL. We will call BELNET the research ISP in the remainder of this section. The mean traffic over the six days period was slightly larger than 32 Mbps, with a one-minute maximum peak at 126 Mbps and a standard deviation of 21 Mbps. The trace begins around 1 AM on a Sunday and finishes six days later around 1 AM also.

The second trace was collected in April 2001 and covers a little less than 5 consecutive days of all the interdomain traffic received by Yucom. Yucom is a commercial ISP that provides Internet access to dialup users through regular modem pools. At that time, the interdomain connectivity of Yucom was mainly provided through high bandwidth links to two transit ISPs. In addition to this transit service, Yucom was also present at the BNIX interconnection point with 15 peering agreements in operation. During the five days of the trace, Yucom received 1.1 terabytes of data. Yucom is representative of an ISP composed of low bandwidth users. We will call Yucom the dialup ISP in the remainder of this section. The trace starts around 8 : 30 AM on a Tuesday and finishes almost 5 days later at midnight. The total traffic also exhibits a daily periodicity with peak hours located during the evening, in accordance with the typical user profile, a dialup user. It had an average total traffic of about 23 Mbps over the measurements, with a one-minute maximum peak at 64 Mbps and a standard deviation of 12 Mbps.

Before analysing the collected traffic statistics, it is useful to have a first look at the BGP table of the studied ISPs. In this section, we assume that the BGP table of both ISPs was stable during the period of the measurements and perform all our analysis based on a single BGP table for each ISP. Using a single BGP table for each ISP is an approximation but since we rely on the BGP table of the studied ISPs our analysis is more precise than other studies [56, 57] that relied on a BGP routing table collected at a different place and time than the packet traces studied in these papers.

The routing table of the dialup ISP contained 102345 active prefixes, covering about 26 % of the total IPv4 address space. This coverage of the total IPv4 address space is similar for the research ISP, with about 24 %, but for 68609 prefixes only. Between late 1999 and mid-2001, 30 % more prefixes are necessary to cover a similar percentage of the IPv4 address space. This has already been analysed elsewhere [58]. Although having different numbers of prefixes in their BGP routing table, the two ISPs cover a similar percentage of the IPv4 address space. This is explained by the average address span per prefix for each ISP, which is about 11000 IP addresses for the dialup ISP and about 15200 addresses for the research ISP. The dialup ISP knew 10560 distinct AS while the research ISP 6298. This difference is mainly due to the large increase in the number of multi-homed sites during the last few years [58]. The average AS path length was 4.2 AS hops for the dialup ISP and 4.5 AS hops for the research ISP.

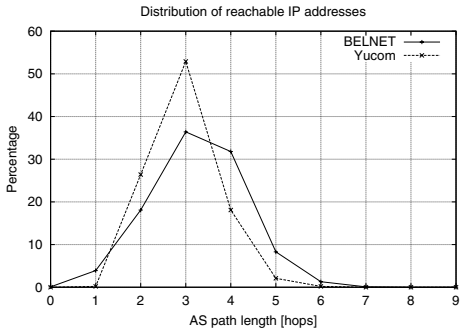


Fig. 12. Distribution of reachable IP addresses.

Figure 12 compares the distribution of the reachable IP addresses for the BGP routing tables of the research ISP and the dialup ISP. The main difference between the two is the more compact distribution for the dialup ISP around a distance of 3 AS hops. The research ISP has its reachable address space more spread over distances of 3 and 4 AS hops. The first 3 AS hops for the dialup ISP provide almost 80 % of the reachable address space while only about 60 % for the research ISP. The difference between the distribution of the reachable IP prefixes seen from the two ISPs is probably due mostly to the 16 months delay between the two traces.

To understand the topological variability of interdomain traffic and the possible levels of aggregation, we consider in this section two different types of interdomain flows. Generally, a flow is defined as a set of IP packets that share a common characteristic. For example, a micro-flow is usually defined as the set of IP packets that belong to the same TCP connection, i.e. the IP packets that share the same source address, destination address, IP protocol field, source and destination ports. In this section, we consider two different types of network-layer flows. A *prefix flow* is the set of IP packets whose source addresses belong

to a given network prefix as seen from the BGP table of the studied ISP. An *AS flow* is defined as the set of IP packets whose source addresses belong to a given AS as seen from the BGP table of the studied ISP. We do not use explicitly the term "flow" to designate traffic coming from a traffic source, but rather the terms "prefix" and "AS" (or "source AS") to denote a *prefix flow* and *AS flow* respectively. Moreover we use the term *order statistics* throughout this paper to denote the traffic flows ordered by decreasing amount of total traffic sent during the all measurements.

Let us first study the amount of aggregation provided by the AS and prefix flows. Figure 13 shows the cumulative percentage of traffic for *order statistics* for prefixes and source AS. On this figure, we have thus ordered the prefixes and AS by decreasing order of the total amount of traffic sent by them over the measurements period, and we have computed their cumulative contribution to the total traffic over the measurements period. The x-axis uses a logarithmic scale to better show the low *order statistics*. Both ISPs seem to have a similar distribution for the most important interdomain traffic sources. The top 100 AS (resp. prefixes) capture 72 % of the total traffic (resp. 52 %) for the dialup ISP while a little less than 60 % (resp. a little more than 40 %) for the research ISP. 90 % of the total traffic is captured by 4.7 % of the AS and by 4.1 % of the prefixes for the dialup ISP. The research ISP required 9.8 % of the AS and 4.5 % of the prefixes to capture 90 % of the total traffic. These results are similar to the findings of earlier studies [59, 60] on the research Internet of the 1970s and the early 1990s. On the other hand, some AS and prefixes contribute to a very small fraction of the total traffic. For the dialup ISP, more than 4000 different AS contributed each to less than 1 megabyte of data during the measurement period and some AS only sent a single packet during this period. For the research ISP, 719 AS sent less than 1 megabyte of data during the six days measurement period.

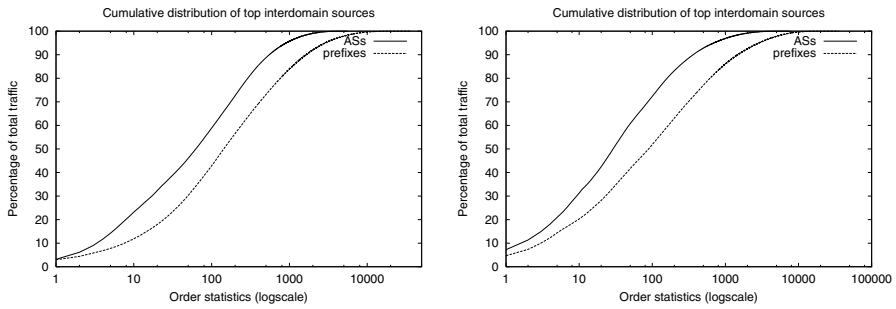


Fig. 13. Cumulative traffic distribution for traffic sources, research ISP (left) and dialup ISP (right).

Another interesting point to mention is that over the measurement period, the research ISP received IP packets from 5606 different AS and 35688 different network prefixes. This corresponds to 89 % of the AS present inside its routing table. Concerning the dialup ISP, it received IP packets from 7668 different AS and 35693 different network prefixes. This corresponds to 72.6 % of the AS present inside its routing table. These figures show that even relatively small ISPs receive traffic from a very large portion of the Internet during a one week period although some sources only send a few packets.

8.1 Interdomain Proximity of the Traffic

The amount of aggregation is not the only issue to be considered when studying interdomain traffic characteristics. Another important issue concerns the topological distribution of the traffic. By topological distribution, we mean the AS distance between the traffic sources and the studied ISP. This distance is important for two reasons. First, usually the performance of an Internet path decreases with the distance between the source and destination AS [61]. Second, if the distance between the source and the destination AS is large, it will be difficult for either the source or the destination to apply mechanisms to control the traffic flow in order to perform interdomain traffic engineering [62].

Another study of the topological distribution of the traffic [63] revealed that the studied ISPs only exchange a small fraction of their traffic with their direct peers (AS-hop distance on 1). Most of the packets are exchanged with ASes that are only a few AS hops away. For the BELNET trace, most of the traffic is produced by sources located 3 and 4 AS hops away while YUCOM mainly receives traffic from sources that are 2 and 3 AS hops away.

8.2 Distribution of the Interdomain Traffic

The previous section showed the amount of traffic generated by interdomain sources for each AS hop distance. Another concern for interdomain traffic engineering is how many sources send traffic at each AS hop distance. Figure 14 presents the cumulative traffic distribution for the top AS for each AS hop distance. In this figure, an AS is not seen as a traffic source from which a flow originates but also as an intermediate node through which a flow passes. This means that an AS located at an AS hop distance of n is seen as the source of the traffic it generates as well as of all the traffic it forwards when considering the AS_PATH information of the BGP routing table. This means that the traffic seen for all AS at an AS hop distance of n contains the traffic originating from all AS hop distances m with $m \geq n$. Because each AS hop distance does not contribute evenly to the total traffic, we have plotted the cumulative traffic percentage for every AS hop distance with respect to the total traffic seen during the measurements period, to show how many AS represent a large fraction of the traffic that crosses the interdomain topology at a given AS hop distance from the local ISP.

The rightmost part of teach curve of Figure 14 shows the uneven distribution of the total traffic among the different AS hop distances.

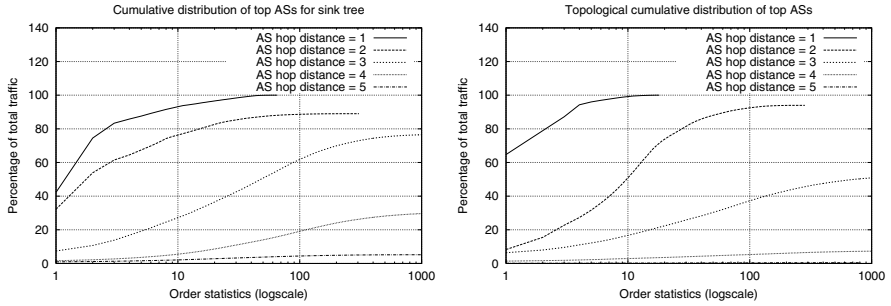


Fig. 14. Cumulative traffic distribution for sink tree, research ISP (left) and dialup ISP (right).

The most important AS at 1 AS hop carries 64 % of the total traffic in the case of the dialup ISP while 42 % for the research ISP. This difference is however lessened when considering the top 3 AS at an AS hop distance of 1, capturing 83 % and 87 % of the total traffic, for the research ISP and the dialup ISP respectively. This shows the predominance of a very small number of BGP peers that provide connectivity for almost all the interdomain traffic of the studied ISPs. At a distance of two AS hops, a few ASes also dominate the traffic with the top 10 carrying more than 77 % of total traffic for the research ISP while 54 % for the dialup ISP. Nevertheless, the traffic produced by AS at a distance of 2 or more AS hops corresponds to 89 % of the total traffic for the research ISP, and 94 % for the dialup ISP. Therefore, a very small fraction of the traffic comes from direct peers themselves. Subsequent distances in terms of AS hops require an increasingly important number of ASes to capture a large fraction of the traffic.

The first AS hop generates 11 % (resp. 6.1 %) of the total traffic, the second AS hop 12.4 % (resp. 42.1 %), the third 46.6 % (resp. 44.4 %), and the fourth 24.9 % (resp. 6.9 %) for the research ISP (resp. for the dialup ISP). The main difference between the two studied ISPs occurs at an AS hop distance of 2. The research ISP has its traffic for the first AS hops that is captured by very few ASes. The dialup ISP on the other hand requires a relatively large number of AS at a distance of 2 AS hops to account for an important fraction of the traffic. This should be compared with the routing table of the dialup ISP (Figure 12) where 52 % of the reachable IP addresses are located at 3 AS hops, and 26 % and 18 % at levels 2 and 4. This means that traffic is unevenly distributed between levels 2 and 4, with more traffic coming from level 2 in comparison to its reachable address space relatively to level 4.

9 BGP-based Interdomain Traffic Engineering

At the interdomain level, ASes have to face various sometimes conflicting issues. On one hand, the traffic is unevenly distributed because BGP seldom takes the right decision on its own and this can cause links to be unevenly loaded and congestion to occur. Moreover, depending on the type of its business, an AS will be more concerned by its incoming or outgoing traffic and thus will use the appropriate traffic engineering techniques. On the other hand, ASes try to maintain as much connections as they can with other ASes for performance and redundancy reasons. If an AS selects a single provider, then all its interdomain traffic will be sent and received from this provider and the only traffic engineering activity will be to balance the traffic if several physical links are used. However, in practice many ASes prefer, for both performance and economical reasons, to select at least two different upstream providers. Since this connectivity is expensive, another concern of ASes will often be to favour the cheapest links.

Moreover, an AS will want to optimise the way traffic enters or leaves its network, based on its business interests. Content-providers that host a lot of web or streaming servers and usually have several customer-to-provider relationships with transit ASes will try to optimise the way traffic leaves their networks. On the contrary, access-providers that serve small and medium enterprises, dialup or xDSL users typically wish to optimise how Internet traffic enters their networks. And finally, a transit AS will try to balance the traffic on the multiple links it has with its peers.

9.1 Control of the Outgoing Traffic

To control how the traffic leaves its network an AS must be able to choose which route will be used to reach a particular destination through its peers. Since an AS controls the decision process on its BGP routes, it can easily influence the selection of the best path. Before looking at the BGP-based techniques later, we first comment the results of an analysis of routing tables collected by Route-Views [64] which show that an hypothetic stub AS connected to two ISPs often receives two routes toward the same destination. The analysis also shows that the selection of a best route in this set is non-deterministic in many cases (because the lengths of the `AS-Path` are equal). Later in this subsection, we briefly describe two techniques that are frequently used to influence the way the traffic leaves the network.

Implications of the BGP Decision Process As shown earlier, the length of the `AS-Path` is used by BGP to rank routes. Some studies [65, 61] have indicated some correlation between the quality of a route and the length of its `AS-Path`. To evaluate the importance of `AS-Path` attribute in the selection of BGP routes, we have simulated the behaviour of a dual-homed stub ISP. For this, we relied on the BGP routing tables collected by route-views [64]. Route-views is a BGP router that maintains multi-hop BGP peering sessions with 20 different ISPs.

We used the BGP routing table recorded on 01 September 2002 at 00:38. We have extracted from the table the routes advertised by each peer of route-views and only consider the peers that advertise their full BGP routing table in this study and used a single BGP peer from each AS.

Based on the BGP routing tables announced by each AS, we have simulated the various possibilities of being dual-homed for a candidate ISP. For this purpose, we performed an experiment with three routers. We used three BGP routers and two BGP sessions. Two routers are used to simulate candidate providers and the third router simulates the multi-homed stub AS. Each of the two candidate AS advertises a BGP routing table from one of the providers. The BGP router of the stub AS runs *GNU Zebra 0.92a* [66] with a default configuration. This implies that this router selects the best route toward each destination advertised by the candidate ASes based on the normal BGP decision process. We modified *Zebra* to allow us collect statistics on the number of routes selected by each criteria of its decision process.

We used this setup to evaluate all possible pairs of upstream providers based on the route-views data. The first result of this evaluation concerns the size of the routing table of the stub ISP. On average, there were 107789 prefixes inside its routing table with a minimum of 95428 and a maximum of 112842. The upper line of figure 15 shows, for each candidate upstream provider, the average number of routes for the 19 experiments where this provider was considered together with another upstream provider. This figure shows that the average number of routes does not vary significantly.

The second element that we considered is the selection of the routes by the BGP decision process of the stub AS. Two cases are possible in our experiment. First, if a route toward a given prefix is only announced by a single candidate AS, this route will automatically be selected. In our experiment, between 87 and 96.5% of the routes received by the stub AS were advertised by both candidate upstream providers. The second line on figure 15 shows, for each candidate upstream provider, the average number of common prefixes between this provider and the other 19 providers. The difference between the routes advertised by different providers can be caused by several factors such as the differences in reachability of each provider and the utilisation of prefix-length filters by some AS as discussed in [67].

The second, and more interesting case to consider is when both candidate providers advertise a route toward each prefix. In this case, the BGP decision process of the stub AS needs to select the best route for each prefix. With the default configuration used by the router of our stub AS, it will first check the `AS-Path` of the received routes. If their `AS-Path` differ, the route with the shortest `AS-Path` will be selected. Otherwise, the tie-breaking rules will be used to select the best route. The bottom line of figure 15 shows, for each candidate upstream provider, the average number of routes from this provider that are selected on the basis of their shorter `AS-Path` by the BGP decision process of our stub AS. For example, concerning AS16150, this figure shows that on average, it advertises 5427 routes with a shorter `AS-Path` than other candidate upstream providers on

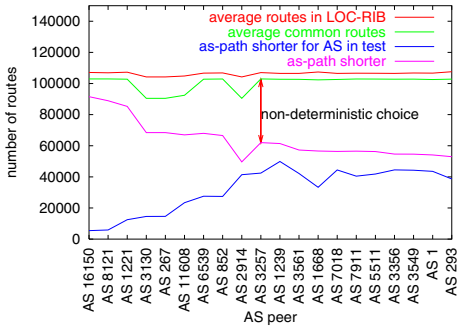


Fig. 15. Quality of the routes announced by an AS — Tests on the 20 peers from Route-Views

an average of 102945 routes in common. This is not surprising since this AS is a much smaller than that the tier-1 ISPs that appear on the right hand side of figure 15.

The second line starting from the bottom in figure 15 shows the average number of routes that were selected by the BGP decision process due to a shorter `AS-Path` in the 19 experiments that involved each candidate AS. For example, concerning AS16150, this line shows that on average, 86104 routes were chosen for their shorter `AS-path` when this provider was confronted with the other providers. The difference between the two bottom lines in figure 15 corresponds to the average number of routes received from the 19 other providers with a shorter `AS-Path` than the considered candidate AS. For AS16150, the other providers advertised on average 80677 routes with a shorter `AS-Path` than this provider. Finally, the difference between the number of common routes and the total average number of shorter routes, shows the average number of routes that were selected in a non-deterministic manner by using the tie-breaking rules of the BGP decision process. For the experiments with AS16150, only 11413 routes were chosen in such a non-deterministic manner.

If we consider a larger AS in figure 15 such as AS1, we find several interesting results. First, on average, AS1 advertises for 43532 prefixes a route with a shorter `AS-Path` than the routes to the same prefixes advertised by any of the other 19 studied ASes. Second, the difference between the bottom line and the line above shows that on average another upstream provider only advertises 10546 shorter routes than AS1. Finally, among all the pairs where AS1 was one of the candidate upstream providers, on average 48473 routes were selected by relying on the tie breaking rules of the BGP decision process. This means that on average 45% of the received routes have the same quality based on their `AS-Path`. A closer look at those common prefixes reveals that 50% of the common prefixes with an `AS-Path` length of three AS-hops are chosen in non deterministic manner. Furthermore, 26.5% of the routes chosen by the tie-breaking rules have an `AS-Path` length of

3 or 4. This indicates that the large ASes advertise short routes towards most destinations.

BGP-based Techniques Two techniques are often used to control the flow of the outgoing traffic. The first technique that can be used by an AS to control its outgoing traffic is to rely on the `local-pref` attribute. This optional BGP attribute is only distributed inside an AS. It can be used to rank routes and is the first criterion used in the BGP decision process (Figure 11). For example, consider a stub AS with two links toward one upstream provider : a high bandwidth and a low bandwidth link. In this case, the BGP router of this AS could be configured to insert a low `local-pref` to routes learned via the low bandwidth link and a higher value to routes learned via the high bandwidth link. A similar situation can occur for a stub AS connected to a cheap and a more expensive upstream provider. In practice the manipulation of the `local-pref` attribute can also be based on passive or active measurements [68]. Recently, a few companies have implemented solutions [69] that allow multi-homed stub ASes and content-providers to engineer their outgoing interdomain traffic. These solutions usually measure the load on each interdomain link of the AS and some rely on active measurements to evaluate the performance of interdomain paths. Based on these measurements and some knowledge of the Internet topology (either obtained through a central server or from the BGP router to which they are attached), they attach appropriate values of the `local-pref` attribute to indicate which route should be considered as the best route by the BGP routers. We will evaluate the impact of `local-pref` by using simulations in section 9.5.

A second technique, often used by large transit ISPs, is to rely on the intradomain routing protocol to influence how a packet crosses the transit ISP. As shown in Figure 11, the BGP decision process will select the nearest IGP neighbour when comparing several equivalent routes received via iBGP. For example, consider in Figure 10 that router R_{27} receives one packet whose destination is R_{45}. The BGP decision process of router R_{27} will compare two routes toward R_{45}, one received via R_{28} and the other received via R_{26}. By selecting router R_{28} as the exit border router for this packet, `AS2` will ensure that this packet will consume as few resources as possible inside its own network. If a transit AS relies on a tuning of the weights of its intradomain routing protocol as described in [70], this tuning will indirectly influence its outgoing traffic.

9.2 Control of the Incoming Traffic

In contrast to the outgoing traffic, it is much more difficult to control the incoming traffic with BGP. Nevertheless access providers can utilise some techniques to influence how the interdomain traffic enters their AS. We first briefly describe these techniques, then we present the simulation model that we used to evaluate one of those techniques and discuss the simulation results in section 9.4.

The first method that can be used to control the traffic that enters an AS is to rely on selective advertisements and announce different route advertisements

on different links. This method suffers from an important drawback: if a link fails, the prefixes that were only announced through the failed link will not be reachable anymore.

A variant of the selective advertisements is the advertisement of more specific prefixes. This technique relies on the fact that an IP router will always select in its forwarding table the most specific route for each packet (i.e. the matching route with the longest prefix). For example, if a forwarding table contains both a route toward `16.0.0.0/8` and a route toward `16.1.2.0/24`, then a packet whose destination is `16.1.2.200` would be forwarded along the second route. This fact can be used to control the incoming traffic by advertising a large aggregate on all links for fault-tolerance reasons and specific prefixes on some links. The advantage of this solution is that if a link fails, the less specific prefix remains available on the other link. Unfortunately, a widespread utilisation of this technique is responsible for a growth of the BGP routing tables. To reduce this growth, many large providers have implemented filters that reject advertisements for too long prefixes [67].

Another method consists in allowing an AS to indicate a ranking among the various route advertisements that it sends. Since the length of the `AS-Path` appears as the second criteria in the BGP decision process, a possible way to influence the selection of routes by a distant ASes is to artificially increase the length of the `AS-Path` of less preferable routes. This is typically done by inserting several times its own AS number in the `AS-Path`. Based on discussions with network operators, it appears that the number of times the own AS is prepended to achieve a given goal can only be found on a trial and error basis.

The last method to allow an AS to control its incoming traffic is to rely on the `multi-exit-discriminator` (MED) attribute. This optional attribute can only be used by an AS multi-connected to another AS to influence the link that should be used by the remote AS to send packets toward a specific destination. It should however be noted that the utilisation of the `MED` attribute is usually subject to a negotiation between the two peering ASes and some ASes do not accept to take the `MED` attribute into account in their decision process. Furthermore, the utilisation of this attribute may cause persistent oscillations [71].

9.3 Simulation Model

The first element of our simulation model is our simulation environment: Javasim [72]. Javasim is a scalable event-driven simulator developed by Hung-Ying Tyan and many others at Ohio-State University. Javasim is written in Java for portability reasons and contains realistic models of various Internet protocols. Although Javasim supports several routing protocols, it did not contain any BGP model. Instead of developing a BGP model from scratch, we choose to port[8] and enhance the BGP implementation developed by B. J. Premore [73] for SSFNet [74]. This model has been extensively validated and tested and has

[8] Our modifications to Javasim are available from `http://www.javasim.org`.

already been used for several simulation studies [75, 76]. We have enhanced it to better support the routing policies that are often used by ISPs as shown earlier.

The second element of our simulation model is the network itself. Since our goal is to evaluate the performance of interdomain traffic engineering, we need a realistic model of the Internet. To evaluate `AS-Path` prepending, we choose to build each AS as composed of a single router that advertises a single IP prefix. This router runs the BGP protocol and maintains BGP sessions with routers in neighbouring ASes. The second element that we needed to specify is the topology of the interdomain links.

An interdomain topology could be obtained from a snapshot of the current Internet, such as the one analysed in [53]. However, a drawback of this approach is that then it is difficult to perform simulations with various topologies to evaluate the impact of the topology on the results. A second method is to rely on a topology built by topology generators. Various topology generators have been proposed and evaluated in the last few years (see [77, 78] and the references therein). It is admitted that two classes of generators can be used: structural and degree-based generators. Structural generators attempt to reproduce the real Internet hierarchy (i.e. tiers, transit ASes and stubs) while degree-based generators approximate a specific property of the real topology, the node degree distribution. It has been shown in [77] that the Internet hierarchy can be better approximated with topologies produced by degree-based generators. Moreover, [78] indicates that degree-based generators are also better suited to approximate the structure of the Internet. Indeed, such generators implicitly create hierarchies closely related to the current Internet hierarchy.

We relied on a degree-based topology generator to produce the various Internet topologies used for the simulations. Our topologies have been generated with Brite [79] which is a highly configurable generator. One of its interesting features is the ability to produce topologies with ASes only, intended to simulate the interdomain level. Brite is able to rely on various mathematical models to generate a topology. We have chosen the Barabasi-Albert model [80] because it is degree-based. This model builds the topology sequentially by adding one AS at a time while relying on two simple principles[81]:

- *Growth*: each node that must be added to the topology is connected to m existing nodes (where m is a parameter of the generator).
- *Preferential Attachment*: when a new interdomain link is created, it connects the AS being added to an existing AS. This AS is selected with a probability which depends on the number of links already attached to each AS. This means that an AS with a lot of interdomain links will be attached to other ASes with a high probability.

A consequence of these two principles is that the ASes which are generated first (i.e. those with a low identifier) have a greater connectivity than the ASes generated last (i.e. those with a large identifier).

In our simulations we use an interdomain topology with two types of ASes. The *core* of the network is composed of a few hundred transit ASes. This core

is generated by using Brite. Note that we do not consider hierarchy in the *core*. For this reason, all *core* ASes all advertise their full routing table to their neighbouring ASes and no routing policies have been defined for the *core* ASes.

In addition to the *core*, our topologies also contain a few hundred stub ASes. Those stub ASes are added to the topology generated by Brite by following a *preferential attachment* principle (the probability for an AS in the *core* to be connected to a stub is function of its current connectivity). Each stub has exactly two connections to two different transit ASes in the *core*. These connections represent *customer-to-provider* links where the stub is the customer and the ASes in the *core* are the providers. We configured BGP policies on the stub ASes to ensure that those ASes do not provide any transit. In the following, we call `lowID` provider (resp. `highID` provider), the provider of the considered stub AS with the lowest (resp. highest) AS number and `lowID` link (resp. `highID` link), the link that leads to this provider.

We have introduced the stub ASes in our network topologies for two reasons. First, they represented 85.6% of the number of ASes on the Internet in October 2002. Second, they will serve as measurement points to evaluate the impact of `AS-Path` prepending on the routes selected by the BGP decision process of each simulated router.

We have performed simulations with several topologies. Due to space limitations, we restrict our analysis to two representative topologies. The first topology is composed of a lightly connected *core* with 200 ASes. This topology was produced by Brite with the value of the m parameter set to 2. 400 dual-homed stub ASes were attached to the *core* ASes with preferential attachment. In total, when considering both the stub and the transit ASes, this topology contains 1594 interdomain links.

The second topology is composed of a dense *core* with 200 ASes. This *core* was produced by Brite with the value of the m parameter set to 4 to model a *core* composed of ASes with a higher connectivity. We have connected 200 dual-homed stub ASes with preferential attachment to this dense core. In total, this second topology contains 1980 interdomain links.

Due to the memory constraints we were not able to perform simulations with more than about 2500 BGP peering sessions. This is one order of magnitude less than the number of interdomain relations reported in [53], but more than one order of magnitude more links than in existing traffic engineering studies.

Another element that should be considered in such a model is the amount of traffic sent by each AS towards each remote AS. In section 8, we have shown that a small number of ASes were responsible for a large fraction of the traffic received by the two studied ISPs. However, those measurement are not sufficient to allow us to determine the behaviour of hundred different ASes and how their traffic would be distributed among the Internet. Developing such a model is outside the scope of this chapter and for the simulations described below, we will consider the interdomain paths between all AS pairs without considering the amount of traffic exchanged.

To evaluate the impact of this BGP traffic engineering technique, we use the stub ASes as measurement points. After a sufficient time to allow the BGP routes to converge, each router sends a special IP packet with the `record route` option toward each remote stub AS. The stub ASes collect the received IP packets and by analysing the `record route` option of each received packets, we can determine all the interdomain paths followed by IP packets. The analysis of all these interdomain paths allows us to study the impact of BGP traffic engineering techniques.

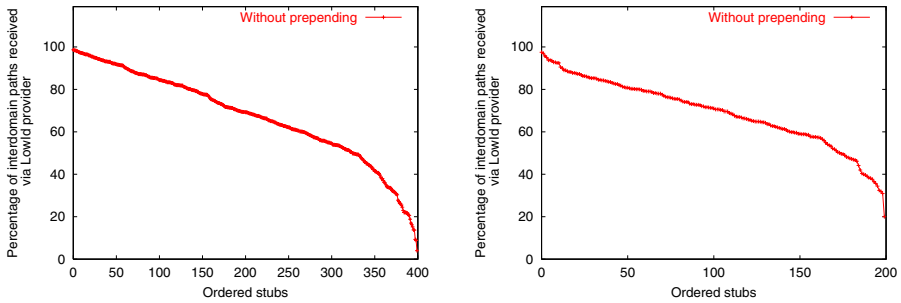


Fig. 16. Distribution of interdomain paths without prepending for lightly (left) connected and dense (right) core

For our first simulation, we configured the stub ASes to send their BGP announcements without any `AS-Path` prepending. Figure 16 shows, for each stub AS, the percentage of the interdomain paths that end on this stub AS and are received via its `lowId` provider. To plot this figure, we have ordered the stub ASes that appear on the x-axis in decreasing percentage of the interdomain paths received via their `lowID` provider. This ordering, determined for each topology, is used for all simulation results described in the remainder of this section.

Several points need to be mentioned concerning Figure 16. First, the distribution of the interdomain paths is not uniform. For both core networks, some stub ASes receive almost all their interdomain packets via one of their providers. The stub ASes that receive almost all their interdomain packets via their `lowID` provider are usually attached to a dense `lowID` provider and a `highID` provider with a very weak connectivity. For the stub ASes that receive almost all their interdomain packets via their `highID` provider, the reason is that this provider is closer to the core ASes with the higher connectivity than their `lowID` provider. Note that with the dense core this situation occurs less often.

A second point to be mentioned is that for the lightly connected core, about 66% of the stub ASes receive more than 60% of their interdomain packets via their `lowID` provider. This is due to the fact that, thanks to its connectivity, the `lowID` provider is, on average, closer to most destinations than the `highID`

provider. For the dense core, results are similar: 72.5% of stub ASes receive more than 60% of their traffic through their `lowID` link.

9.4 Evaluation of `AS-Path` Prepending

As described earlier, `AS-Path` prepending can be used by an ISP to control the flow of its incoming traffic by announcing on some links routes with an artificially long `AS-Path`. Although this technique is used today in the Internet ([65] reports that `AS-Path` prepending affected 6.5 % of the BGP routes in November 2001), there has not been any analysis of its performance to the best of our knowledge.

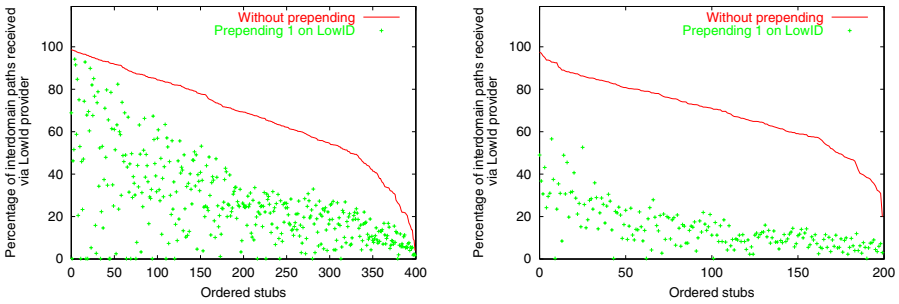


Fig. 17. Distribution of interdomain paths with prepending of 1 on the lowID link for lightly connected (left) and dense (right) cores

To evaluate the impact of `AS-Path` prepending we performed several simulations with the stub ASes configured to send prepended routes to one of their upstream providers. In all simulations, we configured all stub ASes in the same manner to ease comparisons. For our first simulation, each stub AS sent routes with its own AS number prepended once to its `lowID` provider and without prepending to its `highID` provider. In practice, a stub AS could use this prepending to better balance its traffic between its two upstream providers if it receives more traffic via its `lowID` provider.

Figure 17 shows the impact of this prepending on the distribution of the interdomain paths. In this figure, we show the distribution without prepending that was presented in Figure 16 as a reference and use the ordering from this figure to plot the distribution of the interdomain paths with prepending.

The analysis of the simulation with the lightly connected core (Figure 17, left) reveals several interesting results. First, as expected, the distribution of the interdomain paths is affected by the utilisation of `AS-Path` prepending. One can see on Figure 17 that with an `AS-Path` prepending of one on the `lowID` link, the distribution of the interdomain paths has changed for almost all stub ASes. With this amount of prepending, 79% of the stub ASes receive now less than 40% of their interdomain paths via their *lowID* provider.

However, a second important point to mention is that the influence of `AS-Path` prepending is different for each stub AS: some receive all their traffic through the *highID* link while other ASes seem not to be affected. This implies that it can be difficult for a stub AS to predict the impact of the utilisation of `AS-Path` prepending on the distribution of its incoming traffic. This difference is due to the structure of the topology. In our topology as in the Internet, there exists a path between the two upstream providers of a stub AS. The length of the path between these two upstream providers determines the distribution of the interdomain paths after prepending. Let us first consider what happens when the two upstream providers of a stub AS are directly connected. In this case, the `lowID` provider will receive a direct route of two AS-hops and a route of two AS-hops through the `highID` provider. When comparing these two routes, the BGP decision process in our model relies on its random tie-breaker to select one over the other since no routing policies have been configured for *core* ASes. If the BGP decision process of the `lowID` provider selects the route via the `highID` provider, then the stub will receive all the interdomain paths via its `highID` provider. When the two providers are not directly connected, then the impact of the distribution of the interdomain paths depends on their respective connectivity.

In the topology with the dense core, the utilisation of `AS-Path` prepending has a stronger influence on the distribution of the interdomain paths as shown in Figure 17 (right). After prepending, most stub ASes receive less than 40% of their interdomain paths via their `lowID` provider. As with the lightly connected *core*, after prepending some stub ASes do not receive traffic via their `lowID` provider anymore. The difference between the lightly and the dense core topologies can be explained by the connectivity of the providers. In the dense core, there are more direct links between providers and the providers with the lower connectivity have a much better connectivity than the less connected providers in the lightly connected core.

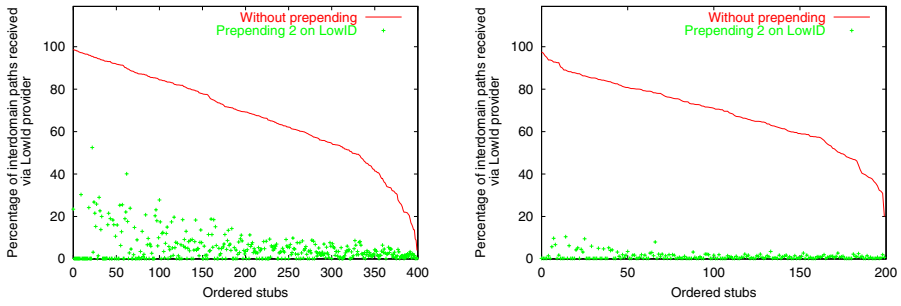


Fig. 18. Traffic distribution after prepending twice on the `lowID` link for the lightly connected (left) and the dense (right) cores

Prepending twice modifies significantly the distribution of the interdomain paths as shown by the simulation results in Figure 18. For the lightly connected core, only 3 stub ASes still receive more than 30% of the interdomain paths via their `lowID` provider after prepending. Furthermore, 86% of the stub ASes receive less than 10% of their traffic through the prepended provider. This means that after prepending twice on the `lowID` link, almost all the interdomain paths have been shifted to the other link. The 3 stub ASes for which the effect is less important have actually a very good connectivity via their `lowID` provider and a very weak connectivity via their `highID` provider. On the dense *core*, prepending twice on the `lowID` link moves almost all the interdomain paths on the `highID` link.

Prepending 7 times, or more, is often used on backup links that should only be used in case of failures. Our simulations with this amount of prepending show that all the interdomain paths are received via the `highID` provider. This is because the topologies we used do not contain routes longer than 6 AS hops. This is similar to the current Internet, where most routes have a length between 2 and 4 AS hops and very few routes a longer than 6 AS hops.

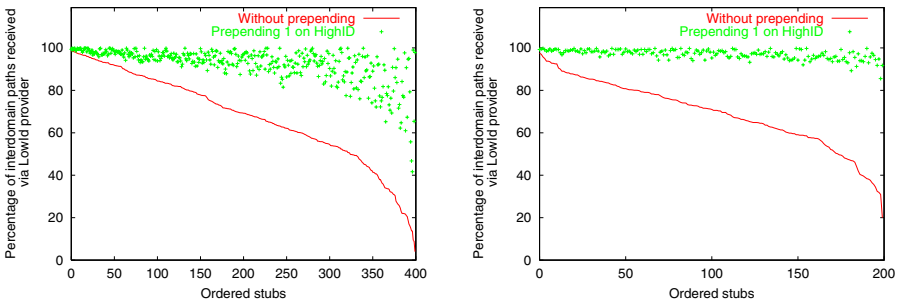


Fig. 19. Distribution of the interdomain paths after prepending once on the `highID` link for the lightly (left) and dense (right) cores

We have also studied the effect of prepending on the link to the `highID` provider. Figure 19 shows that in this case, the distribution of the interdomain paths is much more affected than when prepending was used on the link to the `lowID` provider. This result was expected since the `highID` provider has a weaker interdomain connectivity than the `lowID` provider. As when prepending was used on the `lowID` link, the effect of prepending is not the same for all stub ASes. For the lightly connected *core*, 97 % of the stub ASes receive less than 30% of their traffic through their `highID` provider. A few stub ASes still receive a large part of their interdomain paths via their `highID` link provider despite the prepending. This is due to the good connectivity of the `highID` providers connected to these stub ASes.

For the dense core, the impact of `AS-Path` prepending on the distribution of the interdomain paths is even more important as shown in the right part of Figure 19. Indeed, 84% of the stub ASes receive more than 95% of their interdomain traffic through their lowID provider.

Simulations with larger amounts of `AS-Path` prepending on the `highID` link have shown that most of the interdomain paths are received via the `lowID` provider. For example, for the dense core, all stub ASes receive less than 10 % of the interdomain paths via their `highID` provider when the stub ASes prepend twice the routes announced to this provider. For the dense core, all stub ASes receive less than 2% of the interdomain path via their `highID` provider after prepending twice.

9.5 Influence of `local-pref`

In the previous section, we have studied the impact of `AS-Path` prepending by studying the distribution of the interdomain paths starting from each AS and ending inside each stub AS. This study has been performed by assuming that each source of an interdomain path sends its packets along the best path selected by its decision process. However, as discussed in section 9.1, each AS can easily control its outgoing traffic by using the `local-pref` attribute which is evaluated first in the BGP decision process.

To evaluate the impact of the utilisation of the `local-pref` attribute by the stub ASes, we performed the same simulations as above, but by first configuring `local-pref` on each stub AS to force it to send all its packets via its `lowID` provider. Figure 20 compares the distribution of the interdomain paths in this simulation with the distribution obtained (see Figure 16) when each stub AS sent its packets along its best path toward each destination.

Surprisingly, based on this simulation result, the upstream provider selected by the stub ASes has only a minor influence on the distribution of the interdomain paths. This result is confirmed when we configured each stub AS to send their packets via their `highID` provider as shown in Figure 20 (right). A closer look at the results shows that 68% of the stub ASes receive more than 60% of their interdomain paths through their `lowID` provider when each stub AS sends its packets via its `lowID` provider. On the other hand, 66% of the stub ASes receive more than 60% of the their interdomain paths through their `lowID` provider when each stub AS sends its packets via its `highID` provider. Similar results were obtained with the dense core and when prepending was used.

This result can be explained by analysing all the interdomain paths. A closer look at those paths reveals that a small number of *core* ASes appear in a large fraction of those paths. In Figure 21, we show for the number of appearance of each *core* AS in these interdomain paths. Since each AS sent an IP packet with the `record route` option to each of the other 599 ASes reachable in our topology, there were 358801 interdomain paths. The analysis of those paths reveals that some ASes of the core appear in a very large number of interdomain paths and that many ASes only appear in a small number of paths. In the lightly connected core, one AS appeared in 100031 interdomain paths when the stub

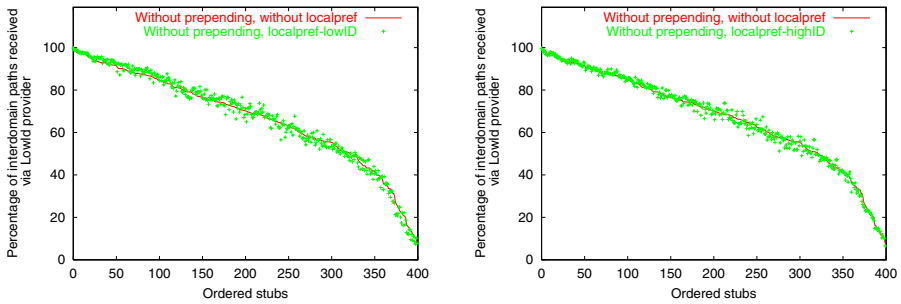


Fig. 20. Impact of `local-pref` on `lowID` link (left) and `highID` link (right)

ASes sent their packets along their best path. This number changed to 109527 (resp. 105236) when the stub ASes sent all their packets via `lowID` (resp. `highID`) provider. Figure 21 shows that the number of interdomain paths passing via each *core* AS does not change significantly when the stub ASes select one upstream provider or another.

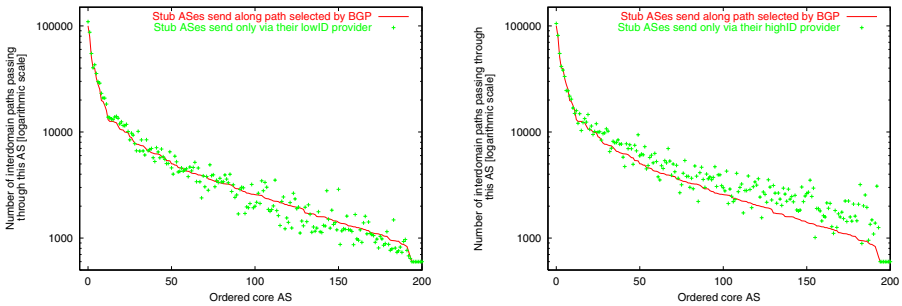


Fig. 21. Impact of the provider selected by the stub ASes on the interdomain paths

Based on these simulation results, it appears that the coupling between the traffic engineering techniques that allow stub ASes to control the flow of their incoming and the outgoing traffic is very weak. This weak coupling implies that the first hops of the path used by a stub AS to send its packets do not significantly influence the last hops of the path that these packets will take to reach their final destination.

10 An Approach for the Propagation of Quality of Service Information

Value-added IP service offerings that are likely to be deployed over the Internet often imply the negotiation of QoS requirements between a customer and a provider, customers possibly being providers themselves. Such QoS information includes parameters like one-way transit delays, inter-packet delay variations, loss rates, *etc.* [82]. This information can also be inferred by a DiffServ Code Point (DSCP, [83]) marking indication, so that a given destination (an IP prefix or a host) could be reachable by different routes, depending on the level of quality both customer and provider have agreed upon.

The enforcement of end-to-end QoS policies is therefore conditioned by the provisioning of resources across domains. The characteristics of these resources will have to comply as much as possible with contractual QoS requirements, which means that the routers involved in the forwarding of the corresponding traffic should be aware of this QoS information, so that it might influence their routing decisions accordingly.

Generally speaking, the BGP attributes that have been specified so far enable the enforcement of a high-level kind of inter-domain routing policy, where service providers can influence the selection of the adjacent domain to reach a given destination. Nevertheless, this is the kind of information that cannot be propagated across domains, and that currently lacks of "QoS-related" knowledge, made of the valued parameters that have been introduced in the beginning of this section.

We believe there is a need for a finer granularity, where service providers would have the ability to exchange information about the classes of service they currently support, the destination prefixes that can be reached by means of traffic-engineered routes that have been computed and selected within their domain, as well as any kind of QoS indication that may contribute to the enforcement of end-to-end QoS policies.

From this perspective, the BGP4 protocol is one of the possible vectors for conveying QoS information between domains. A different way of extending BGP to convey QoS information has been described in [84].

10.1 Propagating Quality of Service Information between Domains

The choice of using the BGP4 protocol for exchanging QoS information between domains is not only motivated by the fact this is currently the only inter-domain (routing) protocol activated in the Internet, but also because the manipulation of attributes is a powerful means for service providers to announce QoS information with the desired level of precision. The approach relies upon the use of an additional attribute, and has identified the following requirements.

First, the approach must be kept scalable. The scalability of the approach can be defined in many ways that include the convergence time to reach a consistent view of the network connectivity, the number of route entries that will have to be

maintained by a BGP peer, the dynamics of the route announcement mechanism (*e.g.*, how frequently and under which conditions should an UPDATE message containing QoS information be sent?), *etc.* Second, the operation of the BGP4 protocol must be kept unchanged. The introduction of a new attribute should not impact the protocol machinery, but the information contained in this attribute may very well influence the BGP route selection process. Third, the approach must allow a smooth migration. The use of a specific BGP attribute to convey QoS information should not constrain network operators to migrate the whole installed base at once, but rather help them in gradually deploying the QoS information processing capability.

The QOS_NLRI Attribute To propagate QoS-related information across domains by means of the BGP protocol, an additional BGP4 attribute, named the QOS_NLRI (QoS Network Layer Reachability Information) attribute is specified in [85].

The QOS_NLRI attribute can be used for two purposes. First, it can be used to advertise a QoS route to a peer. A QoS route is a route that meets one or a set of QoS requirement(s) to reach a given (set of) destination prefixes. Such QoS requirements can be expressed in terms of *minimum one-way transit delay* to reach a destination, the experienced *delay variation* for IP datagrams that are destined to a given destination prefix, the *loss rate* experienced along the path to reach a destination, and/or the identification of the traffic that may use this route (identification means for such traffic include DSCP (DiffServ Code Point) marking). These QoS requirements can be used as an input for the route calculation process embedded in the BGP peers.

Second, the QOS_NLRI attribute can be used to provide QoS information along with the NLRI information in a single BGP UPDATE message. It is assumed that this QoS information will be related to the route (or set of routes) described in the NLRI field of the attribute.

From a service provider's perspective, the choice of defining the QOS_NLRI attribute as an optional transitive attribute is basically motivated by the fact that this kind of attribute allows for gradual deployment of QoS extensions to BGP4: not all the BGP peers are supposed to be updated accordingly, while partial deployment of such QoS extensions can already provide an added value as a means to enforce traffic engineering policies across domains that belong to the same network operator. For example, this would yield QoS-aware BGP route computation and selection for the range of value-added IP services offerings provided by the ISP, as well as an enhanced network resource optimisation and planning within the corresponding domains.

The contents of the QOS_NLRI attribute have been designed so that: *(i)* there is enough room to convey any kind of QoS information. The QoS information that has been identified so far includes transit delay information, loss rate, bandwidth information and Per Hop Behaviour (PHB) identification, *(ii)* the QoS information is tightly related to the destination prefixes that are contained in the NLRI field of the attribute, so as to allow for a route-level granularity

whenever appropriate(instead of an AS-level granularity), as mentioned earlier, as one application example of [86] and *(iii)* The QoS information can be associated to other network protocols than IPv4, including its version 6 [87] whose header format includes implicit traffic engineering capabilities that should allow for an even finer level of granularity.

Basic Operation The QOS_NLRI attribute is conveyed in BGP UPDATE messages that are exchanged between peers of different domains. BGP routers that are capable of processing the information contained in the QOS_NLRI attribute can provide this indication to their peers by means of the Capabilities Advertisement mechanism, as defined in [88].

By *processing* the QoS information contained in the QOS_NLRI attribute, we basically mean two different operations. First, the QoS information can be altered as long as it crosses the Internet. For example, one-way transit delay information can be modified by means of additive operation, whenever the information is propagated hop-by-hop between domains. And second, the QoS information can actually influence the BGP route selection process.

Because it is an optional and transitive attribute, the QOS_NLRI attribute will be propagated across domains, even though there may be peers along the path that are unable to process the information conveyed in the attribute. In this case, the Partial bit of the attribute will be set by such peers, meaning that the information propagated by the attribute is incomplete.

10.2 Simulation Work

The simulation work aims at validating the technical feasibility of the approach (hence qualifying the added value introduced by the use of the QOS_NLRI attribute). It is also intended to focus on the scalability of the approach, within the context of the enforcement of inter-domain traffic engineering policies.

The simulation work has been organised as a step-by-step approach, which consists in the following four phases.

First, an IP network composed of several autonomous systems is modelled. Since this simulation effort is primarily focused on the qualification of the scalability related to the use of the QOS_NLRI attribute for exchanging QoS-related information between domains, it has been decided that the internal architecture of such domains be kept very simple, i.e. without any specific IGP interaction

Second, inside this IP network, some BGP peers that are QOS_NLRI aware, i.e. they have the ability to process the information conveyed in the attribute, while the other routers do not recognise the QOS_NLRI attribute by definition. Those routers will forward the information to other peers, by setting the Partial bit in the attribute, meaning that the information conveyed in the message is incomplete. This approach allows to elaborate on the added value introduced by a gradual deployment of the QoS extensions to BGP4.

Third, as far as QOS_NLRI aware BGP peers are concerned, they will process the information contained in the QOS_NLRI attribute to possibly influence

the route decision process, thus yielding the selection (and the installation) of distinct routes towards a same destination prefix, depending on the QoS-related information conveyed in the QOS_NLRI attribute. From this implementation perspective, the BGP routing tables have been modelled so that they contain a "sub-section" where QOS_NLRI-capable peers will store the information conveyed in the attribute.

The last phase is to modify the BGP route decision process: at this stage of the simulation, the modified decision process relies upon the one-way delay information and it also takes into account the value of the Partial bit of the attribute.

Once the creation of these components of the IP network has been completed (together with the modification of the BGP route selection process), the behaviour of a QOS_NLRI-capable BGP peer is as follows. Upon receipt of a BGP UPDATE message that contains the QOS_NLRI attribute, the router will first check if the corresponding route is already stored in its local RIB, according to the value of the one-way delay information contained in both QoS Information Code and Sub-code fields of the attribute.

If not, the BGP peer will install the route in its local RIB. Otherwise (i.e. an equivalent route already exists in its database), the BGP peer will select the best of both routes according to the following criteria:

If both routes are said to be either incomplete (Partial bit has been set) or complete (Partial bit is unset), the route with the lowest delay will be selected, Otherwise, a route with the Partial bit unset is always preferred over any other route, even if this (complete) route reflects a higher transit delay.

If ever both Partial bit and transit delay information are not sufficient to make a decision, the standard BGP decision process (according to the breaking ties mechanism depicted in [5]) is performed.

A Case Study The current status of the simulation work basically relies upon the one-way transit delay information only, as well as the complete/incomplete indication of the Partial bit conveyed in the QOS_NLRI attribute.

The IP network has been modelled so that it is composed of 6 autonomous systems and 11 BGP peers. Future scenarios will be composed of more ASs. Figures /reffigure:case-study depicts the actual processing of the QoS-related information conveyed in the QOS_NLRI attribute, depending on whether the peer is QOS_NLRI-aware or not.

Figure 22 depicts the modelled network for a first case study. In this figure, the routers marked with an asterisk support the QOS_NLRI attribute while the others do not and the arrows indicate the delays known by each QOS enabled router. Let us consider the propagation of a BGP UPDATE message that contains the QOS_NLRI attribute, in the case where the contents of the attribute are changed, because of complete/incomplete conditions depicted by the Partial bit of the QOS_NLRI attribute.

Router S is a QOS_NLRI-capable speaker. Assume that it takes 20 milliseconds to node S to reach network 192.0.20.0: this information will be conveyed in

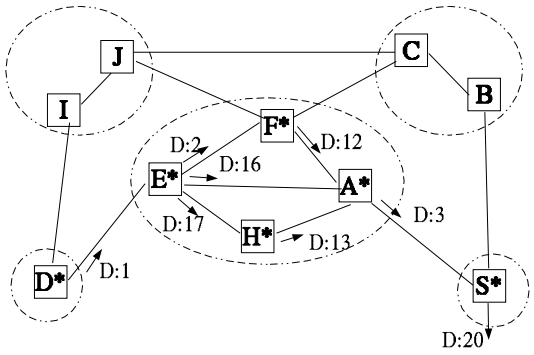


Fig. 22. A case study

a QOS_NLRI attribute that will be sent by node S in a BGP UPDATE message with the Partial bit of the QOS_NLRI attribute unset.

Router A is another QOS_NLRI BGP peer, and it takes 3 milliseconds for A to reach router S. Node A will update the QoS-related information of the QOS_NLRI attribute, indicating that, to reach network 192.0.20.0, it takes 23 milliseconds. Router A will install this new route in its database, and will propagate the corresponding UPDATE message to its peers.

On the other hand, router B is not capable of processing the information conveyed in the QOS_NLRI attribute, and it will therefore set the Partial bit of the QOS_NLRI attribute in the corresponding UPDATE message, leaving the one-way delay information detailed in both QoS Information Code and Sub-code unchanged.

Upon receipt of the UPDATE message sent by router A, router E will update the one-way delay information since it is a QOS_NLRI-capable peer. Finally, router D receives the UPDATE message, and selects a route with a 40 milliseconds one-way delay to reach network 192.0.20.0.

This example shows that the selection of a delay-based route over a BGP route may not yield an optimal decision. In the above example, the 40 ms-route goes through routers D-E-A-S, while a "truly optimal" BGP route would be through routers D-E-F-A-S, hence a 38 ms-route. This is because of the iBGP rule that does not allow router F to send an UPDATE message towards router E, because router F received the UPDATE message from router A thanks to the iBGP connection it has established with A.

These basic observations confirm that the enforcement of a QoS policy between domains by using the BGP4 protocol is obviously conditioned by the BGP4 routing policies that are enforced within each domain.

Preliminary Simulation Results Figure 23 reflects the first results obtained from a simulation network composed of 9 autonomous systems and 20 BGP peers. It shows the impact of the introduction of QOS_NLRI-capable peers in

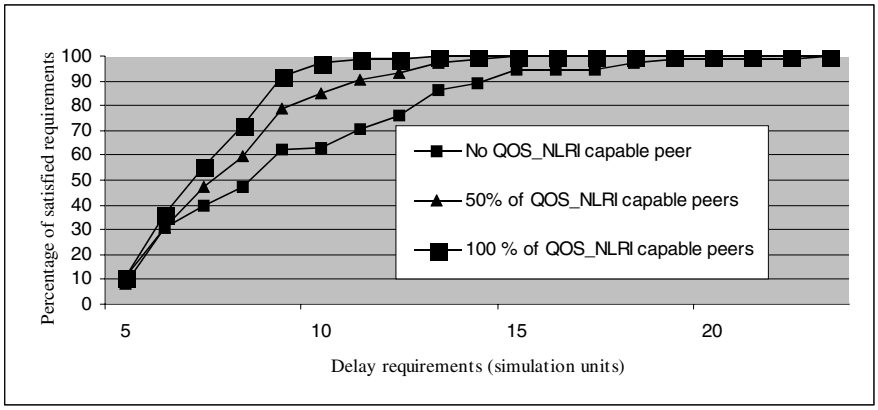


Fig. 23. Preliminary results of QOS_NLRI-capable peers.

the network, where the enhanced route selection process results in an increase of the percentage of satisfied QoS requirements, especially in the region where these requirements are stronger (simulation units are arbitrary units so as to provide a better readability of the chart. In practice, such units would be expressed in milliseconds if the one-way transit delay were the key differentiator for route computation). Furthermore, the chart clearly shows that the efficiency of a QOS_NLRI-inferred BGP route selection process is critical for the strongest requirements (e.g. for real-time traffic) at the sole expense of introducing a new attribute (and possibly influencing the breaking ties' mechanism logic), without questioning the BGP-based exchange of information between the domains of the Internet, as it's been done for several decades.

11 Improving Communication Qualities through Multi-homing

Since global connectivity has become a critical resource for organisations, sites attempt to improve the qualities of their communication facilities obtaining Internet connectivity through two or more providers. However, actual benefit obtained depends on how end-hosts and the routing system cope with this multiple path information, i.e. the multi-homing solution available. In the IPv4 Internet, several multi-homing solutions have been deployed, providing different levels of benefit. While some of them can be directly adopted in the IPv6 Internet, others do not seem to be aligned with some key design criteria of IPv6, such as routing system scalability or aggressive route aggregation, making them unsuitable for direct adoption. In this section we will review and evaluate existent and proposed solutions for the improvement of communication qualities through multi-homing.

11.1 Currently Available Solutions

In this section we will describe currently available multi-homing solutions. All the solutions described are available for IPv4 and while it is also possible to deploy them for IPv6, some of them are deemed unacceptable since they would jeopardise routing system scalability features.

The Incumbent Solution: Indiscriminate Route Injection The most widely deployed multi-homing solution in IPv4 is based on the announcement of the site prefix through all its providers, as illustrated in figure 24.

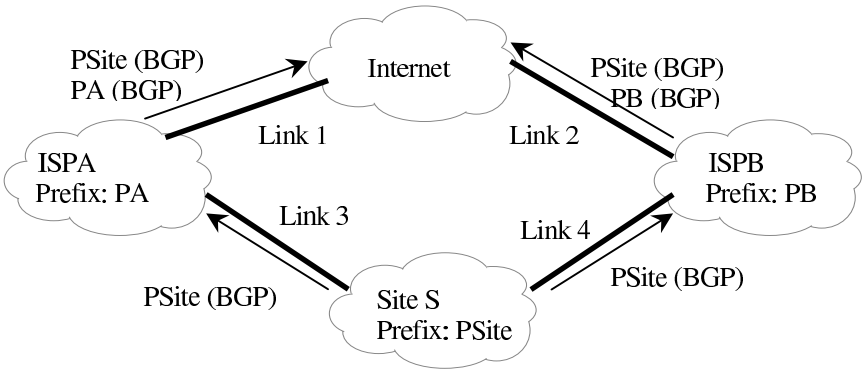


Fig. 24. Unrestricted route injection

In this solution, the site S obtains a prefix assigned directly from the RIR (Regional Internet Registry) or from one of its providers (ISPA or ISPB). Then, the site announces this prefix (PSite) to its providers using BGP. ISPA and ISPB announce the prefix to its providers, and this process continues until the route is announced into the Default Free Zone (DFZ). It must be noted that the PSite prefix can be part of one provider aggregate or it can be obtained directly from a RIR, although in any case the PSite prefix has to be announced separately in order to avoid that the longest prefix match rule diverts all the traffic to the provider announcing the more specific route.

This mechanism presents many desirable properties, such as optimal fault tolerance capabilities that includes preserving established connections throughout an outage, since alternative routes are managed without end-node perception. However, each multi-homed site using this solution contributes with two routes to the DFZ routing table, imposing additional stress to already over-sized routing tables. A resulting concern is the growing time taken for route withdrawal, that depending on the case can be long enough to cause retransmission from end-hosts [89]. So, this solution that originally provided optimal functionality and preserved almost unchanged established communications, currently is presenting

poor performance because of its own success. For this reason, this solution is not considered to be acceptable for IPv6, for which some sort of provider aggregation is to be adopted, as it is described next.

Available Solutions Compatible with the Provider Aggregation Scheme In this section we will present solutions that are compatible with the usage of provider aggregatable addresses. We will first describe the configuration of a multi-homed site with provider aggregatable addresses and then we will present several current available multi-homing solutions compatible with this configuration.

Provider Aggregatable Addresses and Multi-homed Sites In order to reduce the routing table size, the usage of some form of provider aggregation is recommended, meaning that sites obtain prefixes which are part of the allocation of their provider, so that its provider only announce its own aggregate. In this scheme, the most beneficial aggregation is achieved by aggregating end-site prefixes into the prefix allocated to their direct provider [90], so that direct provider aggregation of end-sites is deemed necessary for scalability. Further aggregation, i.e. the aggregation of provider prefixes into their upstream provider prefix, leads to moderate aggregation benefits but it presents deployment challenges, making its adoption uncertain.

When provider aggregation of end-site prefixes is used, end-site host interfaces obtain one IP address from each allocation in order to be able to receive traffic through both providers. Note that ISPs only announce their aggregates to their providers, and they do not announce prefixes belonging to other ISPs aggregates.

This configuration presents several concerns:

- Additional address consumption, which may or may not be an issue depending on the protocol version used.
- Increased connection establishment time in case of failure. When Link 1 or Link 3 becomes unavailable, addresses containing the *PASite* prefix are unreachable from the Internet. New incoming (from the site perspective) connections addressed to *PASite* addresses will suffer from an increased establishment time, since the connection request to the unavailable address will timeout and alternative address, containing PBSite prefix will be used.
- Established connections will not be preserved in case of an outage. If Link 1 or Link 3 fails, already established connections that use addresses containing the *PASite* prefix will fail, since packets addressed to the *PASite* aggregate will be dropped because there is no route available for this destination. Note that an alternative path exists, but the routing system is not aware of it.
- Ingress filtering incompatibility. Ingress filtering is a widely used technique for preventing the usage of spoofed addresses. However, in this configuration, additional difficulties arise from the interaction between source address selection mechanism and intra-site routing systems, since the packet exit path

and the source address carried in the packet must be coherent, in order to bypass ingress filtering mechanisms.
- Source address selection difficulties. When Link 1 or Link 3 fails, site hosts should not use addresses containing *PASite* prefix for initiating external communications, since reply packets will be dropped because there is no route available to the source address. If Link 3 is the one that has failed, the site exit router connecting with ISPA can be aware of the outage and it can propagate the information to the hosts, so that the unavailable addresses is no longer used (this can be done using Router Advertisement [91] and Router Renumbering [92]). However, if Link 1 is the unavailable one, the situation cannot be solved using only these tools.

In brief, this configuration present better availability than the single-homed configuration but it does not improve the qualities of established connections during an outage. Therefore, additional mechanisms are needed to allow the usage of provider aggregatable addresses while still obtaining benefits equivalent to the incumbent multi-homing solution.

Auto-route Injection A multi-homing mechanism compatible with provider aggregation is presented in [93] and it is based on the outage-triggered injection of routes. In normal operation, the site S, that obtains one address block per provider, announces via BGP *PASite* to ISPA and *PBSite* to ISPB. In case of an outage, which can be detected by S by comparing the routing announcement obtained from both ISPs, the site injects both prefixes to the still working ISP.

This solution preserves established communications in any failure mode in which at least one path is available, and also allows new communications to be established using all advertised addresses. The impact of the outage in communications depends on the location of the outage. If the outage is topologically close to the site, route injection will be fast and routing system can converge fast enough so that the communications remains unaware of the failure. However, as the topological distance from the site to the faulty device increases, the time needed for route injection and routing system convergence grows, affecting communication. The impact will be an increased delay, which can even imply retransmissions from the source.

The presented mechanism preserves aggregation during normal operation and it only injects additional routing information during outages. However, considering the size of the Internet, probably there will be many simultaneous outages at any given time, so that additional routing information will always be present in the global routing tables. This solution is already deployed in IPv4 and it is susceptible to be deployed in IPv6. Whether the amount of extra information is acceptable or not for IPv6 remains to be evaluated.

Multi-homing Support at Site Exit Routers. This mechanism, presented in [93] and developed for IPv6 in [94], is aimed to provide last-hop link fault tolerance which is achieved through tunnels between site egress routers (RA, RB) and ISP border routers (BRB, BRA) as it is depicted in figure 25. In normal operation

tunnelled paths are advertised with a low priority, to guarantee that traffic is routed through direct links whenever possible. In case of link failure, the link that is down is no longer advertised as a valid route, so the tunnelled path becomes the preferred route in spite of its low priority.

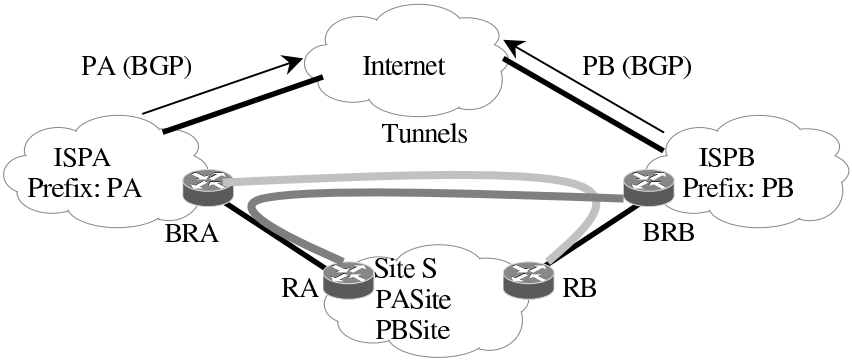


Fig. 25. Multi-homing support at site exit routers

This mechanism has already been implemented in both IPv4 and IPv6 and it is fully compatible with the provider aggregation scheme. In this case, the solution provides better response times than the previous one in the case that the outage is caused by a direct link failure. New connections can still be established using all addresses, and established connections are preserved. The only impact perceived by end-hosts is an increase of the transit delay due to the extra hops introduced, and a reduction of the path MTU that can originate fragmentation or packet loss.

The main drawback is the limited fault tolerance achieved by the solution, since it only preserves connectivity through a direct link failure episode, while other faults such as ISP breakdown remain uncovered.

11.2 Alternative Multi-homing Solutions

In this section we will present multi-homing solutions that have been proposed but that have not been widely accepted yet. Some of these mechanisms are IPv6-only, since they lie on special IPv6 features, such as the extended address range. Others have been deployed for IPv6 but also apply for IPv4. We will classify the presented solutions into two groups: host centric mechanisms and router centric mechanisms.

Host Centric Solutions Host centric solutions are mechanisms residing on hosts that deal essentially with address management to provide multi-homing

capabilities. So, the host is capable of discovering multiple alternative addresses of the other end and it is also capable of discerning when to use them. In host centric solutions, interaction with routing system may provide additional information so that if it is available, the multi-homing mechanism performance improves, but if it is not, the solutions still work properly. Host solutions can be implemented at three levels, namely network level, transport level and application level, as it will be presented next.

Network Level Solutions A host network level solution is presented in [95]. In this solution, identifier and locator functions of IP addresses are explicitly separated by defining a new type of address, called *LIN6 generalised ID*, which is constructed concatenating a reserved prefix (the LIN6 prefix) with the Interface Identifier part of IPv6 addresses. These addresses are used by transport and upper layers for identifying endpoints of communications, but they are never transmitted in packets, since they are mapped into regular routable addresses by the LIN6 module of the host network layer. Routable addresses are formed by concatenating regular network prefixes delegated by ISPs and the Interface Identifier of the interfaces.

LIN6 generalised Identifiers are stored in the DNS and mapping information is stored in Mapping Agents whose location can be found in newly defined DNS records. Fault tolerance is supported by changing the routable address to which the LIN6 generalised ID is mapped to, i.e. when an ICMP error message is received, an alternative routable address is used for the same LIN6 generalised ID. It should be noted that the IP layer is not connection oriented and no connection timeout available to indicate that an error has occurred, so the only available mechanism for fault detection is based on ICMP error messages.

Transport Level Solutions Transport layer solutions propose that a transport layer connection can be identified not just by one IP address on each end, but by a set of addresses on each end. An extension to TCP has been proposed in [96], so that SYN packets carry several IP addresses available to reach the source. This information can be transported either in a TCP option or in a new IPv6 Extension Header. Fault tolerance is provided by switching addresses used to reach the other end, and address switching is triggered by TCP connection timeout.

A more sophisticated transport level solution is provided by the Stream Control Transport Protocol (SCTP) [97]. In this case, several IP addresses can be also used to reach the other end of the communication, but more sophisticated fault tolerance support is available based on a heartbeat mechanism to probe correspondent endpoint reachability through every address. This improved mechanism enables an intelligent choice of the alternative address to use.

Application Level Solutions Another possible approach is to delegate multihoming support to the application level. This implies that applications must deal with multiple addresses per endpoint i.e. they must be able to recognise

that packets coming from different source addresses belong to the same communication. This provides maximum flexibility to applications. However, this also imposes extra complexity to application developers, who must deal with routing and reachability issues, being this not always desirable.

Final Considerations on Host Centric Solutions Support for fault tolerance in host centric solutions is based on retransmissions i.e. when an outage occurs, packets are lost and optionally ICMP errors packets are sent or timeouts occur, so the packet is retransmitted with an alternative address. In this type of solutions, all failure modes are covered, since whenever a path is available through one of the advertised addresses, the communication will be preserved. However, the host becomes aware of the outage at some communication layer, since packets must be retransmitted, introducing an increased delay in certain packets. It should be noted that this delay will probably be higher than the one introduced by the previous solution, in the case of a direct link outage.

Router Centric Solutions Currently deployed solutions are essentially router centric solutions i.e. routers provide the capabilities needed for multi-homing support while hosts remains unaware. This type of solutions has already proven to provide good fault tolerance, but it also has exhibited scalability limitations. Alternative mechanisms have been proposed which do not introduce additional information into the global routing table and therefore preserve aggregation. The Multi-Homing Aliasing Protocol (MHAP) [98] is a router mechanism that proposes the usage of two address spaces, one for multi-homed endpoint identification and another one for routing. In MHAP, multi-homed hosts are identified by a special type of IPv6 addresses, called *Multi-homed addresses* which are formed by a prefix specially reserved for these purposes concatenated with the corresponding Interface Identifier. However, these addresses are not used to route the packet through the Internet, but they are translated into regular Provider Aggregatable addresses. So the typical communication between a single-homed source and a multi-homed destination would be as follows: The source obtains the multi-homed address of the destination host through a DNS query. Then it sends the packet with the obtained multi-homed as destination address. When the packet reaches the site exit router, a process within the router called the *MHTP client* obtains the Provider Aggregatable addresses, and switches the multi-homed address with one of the obtained addresses. Afterwards, the packet is routed to the destination site using this address. Finally, the ingress router at the destination site translates the destination address back to the original multi-homed one. In this scheme, traffic addressed to multi-homed sites is translated twice, so that end-hosts remain unaware of changes. Mapping information about multi-homed addresses and Provider aggregatable addresses are exchanged on demand by concerned routers using BGP. Fault tolerance capabilities are provided by exchanging keepalives between both end-site exit routers. If keepalives are not received, an alternative address is used. It should be noted that keepalive timeout can be tuned so that established communications do not timeout.

12 Summary and Conclusions

In this chapter, we have discussed several Internet traffic engineering techniques. We have first addressed the needs of single autonomous systems by using Diffserv and MPLS. We have proposed an automated provisioning system, targeting to support demanding traffic requirements (SLSes), while at the same time optimising the use of network resources. We seek to place the traffic demands to the network in such a way as to avoid overloading parts of the networks and minimise the overall network cost. We devised a non-linear programming formulation and we proved though simulation that we achieve our objectives.

Moreover, we presented how this intra-domain traffic engineering and provisioning system can be policy-driven. We described the components of the necessary policy-based system extensions that need to be deployed in order to enhance or modify the functionality of the policy influenced components reflecting high-level business decisions. We then enlisted the policies which can be used to influence the behaviour of Network Dimensioning and presented an example of the enforcement of such a policy.

In the second part of this chapter we have discussed the traffic engineering techniques that are applicable between autonomous systems. We have first described the behavior of BGP and explained several techniques that can be used to control the flow of interdomain traffic. We have also discussed the characteristics of interdomain traffic and have shown that although an AS will exchange packets with most of the Internet, only a small number of ASes are responsible for a large fraction of the interdomain traffic. This implies that an AS willing to engineer its interdomain could move a large amount of traffic by influencing a small number of distant ASes. Second, the sources or destinations of interdomain traffic are not direct peers, but they are only a few ASes hops away. This implies that interdomain traffic engineering solutions should be able to influence ASes a few hops beyond their upstream providers or direct peers.

We have then presented a detailed evaluation of techniques that can be used to control the flow of the incoming traffic. Our detailed simulations of `AS-Path` prepending has shown that it is difficult to utilize this technique to achieve a given goal. Our simulations with `local-pref` have shown that the utilization of this technique had only a small influence on the traffic received by remote stub ASes.

We have then described the QoS_NLRI attribute that can be used to distribute QoS information between domains and have shown preliminary simulation results. However, this BGP-based approach to inter-domain traffic engineering raises issues that remain un-addressed, like the scalability of the approach and the QoS route aggregation capabilities of enhanced BGP peers.

Finally, we have discussed IPv6 multi-homing solutions. A set of current multi-homing solutions have been presented, starting with currently deployed ones and then presenting alternative new approaches that attempts to provide multi-homing benefits without jeopardizing overall scalability. We have then analyzed the improvements that each one of the described solutions provides from the communications point of view. As a conclusion, no solution that has been

proposed yet combines scalability with the same benefits as the incumbent solution, although the host-centric approach could be an acceptable one.

References

[1] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. *Overview and Principles of Internet Traffic Engineering*. IETF, Informational RFC-3272, May 2002.

[2] S. Blake and et al. *An Architecture for Differentiated Services*. IETF, Informational RFC-2475, December 1998.

[3] E. Rosen, A. Viswanathan, and R. Callon. *Multi-protocol Label Switching Architecture*. IETF, Standards Track RFC-3031, January 2001.

[4] M. Sloman. Policy Driven Management For Distributed Systems. *Journal of Network and Systems Management*, 2(4):333–360, December 1994.

[5] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). Internet Engineering Task Force, RFC 1771, March 1995.

[6] Internet Engineering Task Force (IETF). Traffic Engineering Working Group (tewg). information available at:.

[7] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus. *Requirements for Traffic Engineering Over MPLS*. IETF, Informational RFC-2702, September 1999.

[8] F. Le Faucheur and W. Lai (eds.). *Requirements for support of Diff-Serv-aware MPLS Traffic Engineering*. Internet draft, <draft-ietf-tewg-diff-te-reqts-07.txt>, work in progress, February 2003.

[9] D. Katz, K. Kompella, and D. Yeung. *Traffic Engineering Extensions to OSPF Version 2*. IETF Internet draft, <draft-katz-yeung-ospf-traffic-10.txt>, work in progress, June 2003.

[10] A. Feldman and J. Rexford. IP Network Configuration for Intra-domain Traffic Engineering. *IEEE Network Magazine*, 15(5):46–57, September/October 2001.

[11] P. Aukia, M. Kodialam, P. V. Koppol, T. V. Lakshman, H. Sarin, and B. Suter. RATES: A Server for MPLS Traffic Engineering. *IEEE Network Magazine*, 14(2):34–41, September/October 2000.

[12] B. Fortz and M. Thorup. Internet Traffic Engineering by Optimizing OSPF Weights. In *Proc. of IEEE INFOCOM'00*, pages 519–528. Israel, March 2000.

[13] M. Kodialam and T. V. Lakshman. Minimum Interference Routing with Applications to Traffic Engineering. In *Proc. of IEEE INFOCOM'00*, pages 884–893. Israel, March 2000.

[14] P. Trimintzios, I. Andrikopoulos, G. Pavlou, P. Flegkas, P. Georgatsos D. Griffin, D. Goderis, Y. T'Joens, L. Georgiadis, C. Jacquenet, and R. Egan. A Management and Control Architecture for Providing IP Differentiated Services in MPLS-based Networks. *IEEE Communications Magazine*, 39(5):80–88, May 2001.

[15] K. Nichols, V. Jacobson, and L. Zang. *A Two-bit Differentiated Services Architecture for the Internet*. IETF, Informational RFC-2638, July 1999.

[16] J. De Clercq, S. Van den Bosch, and A. Couturier. *An Architecture for a Gradual Deployment of end-to-end QoS on an Internet-wide Scale*. IETF Internet draft, <draft-declercq-vsn-arch-01.txt>, work in progress, June 2003.

[17] P. Pan, Hahne E, and H. Schulzrinne. BGRP: A Tree-Based Aggregation Protocol for Inter-domain Reservations. *Journal of Communications and Networks*, 2(2):157–167, June 2000.

[18] E. Mykoniati, C. Charalampous, P. Georgatsos, T. Damilatis, D. Goderis, P. Trimintzios, G. Pavlou, and D. Griffin. Admission Control for Providing QoS in IP DiffServ Networks: the TEQUILA Approach. *IEEE Communications Magazine*, 41(1):38–44, January 2003.
[19] G. Ventre (ed.). *Chapter 8: Engineering of Future Internet Services (this book).* Springer, 2003.
[20] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms and Applications.* Prentice Hall, 1993.
[21] D. Mitra and K. G. Ramakrishnan. A Case Study of Multi-service, Multi-priority Traffic Engineering Design for Data Networks. In *Proc. IEEE GLOBECOM'99*, pages 1087–1093. Brazil, December 1999.
[22] D. Mitra, J. A. Morrison, and K. G. Ramakrishnan. Virtual Private Networks: Joint Resource Allocation and Routing Design. In *Proc. IEEE INFOCOM'99*, pages 884–893. USA, March 1999.
[23] S. Van den Bosch, F. Poppe, H. De Neve, and G. Petit. Choosing the Objectives for Traffic Engineering in IP Backbone Networks based on Quality-of-Service Requirements . In *Proc. of 1st Workshop on Quality of future Internet Services (QofIS'00)*, pages 129–140. Germany, September 2000.
[24] S. Suri, M. Waldvogel, and P. R. Warkhede. Profile-Based Routing: A New Framework for MPLS Traffic Engineering. In *Proc. of 2nd Workshop on Quality of future Internet Services (QofIS'01)*, pages 138–157. Portugal, September 2001.
[25] D. Mitra and Q. Wang. Stochastic Traffic Engineering, with Applications to Network Revenue Management. In *Proc. of IEEE INFOCOM'03*. USA, March/April 2003.
[26] N. Duffield, P. Goyal, A. Greenberg, P. Mishra, K.K. Ramakrishnan, and J. Van der Merwe. A Flexible Model for Resource Management in Virtual Private Networks. In *Proc. of ACM SIGCOMM'99*, Massachussetts, USA, August/September 1999.
[27] A. Kumar, R. Rastogi, A. Silberschatz, and B. Yener. Algorithms for Provisioning Virtual Private Networks in the Hose Model. In *Proc. of ACM SIGCOMM'01*, San Diego, USA, August 2001.
[28] A. Jüttner, I. Szabó, and Á Szentesi. On Bandwidth Efficiency of the Hose Resource Management Model in Virtual Private Networks. In *Proc. of IEEE INFOCOM'03*. USA, March/April 2003.
[29] Z. Wang, Y. Wang, and L. Zhang. Internet Traffic Engineering without Full Mesh Overlaying. In *Proc. IEEE INFOCOM'01*. Alaska, April 2001.
[30] Y. Breitbart, M. Garofalakis, A. Kumar, and R. Rastogi. Optimal Configuration of OSPF Aggregates. In *Proc. IEEE INFOCOM'02*. USA, June 2002.
[31] P. Van Mieghem (ed.). *Chapter 3: Quality of Service Routing (this book).* Springer, 2003.
[32] S. Chen and K. Nahrstedt. An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions. *IEEE Network Magazine*, 12(6):64–79, November/December 1998.
[33] A. Elwalid, C. Jin, S. H. Low, and I. Widjaja. MATE: MPLS Adaptive Traffic Engineering. In *Proc. IEEE INFOCOM'01*, pages 1300–1309. Alaska, April 2001.
[34] A. Sridharan and R. Guérin. Achieving Near-Optimal Traffic Engineering Solutions for Current OSPF/IS-IS Networks. In *Proc. of IEEE INFOCOM'03*. USA, March/April 2003.
[35] Z. Cao, Z. Wang, and E. Zegura. Performance of Hashing-based Schemes for Internet Load Balancing. In *Proc. of IEEE INFOCOM'00*, pages 332–341. Israel, March 2000.

[36] P. Trimintzios, P. Flegkas, and G. Pavlou. Policy-driven Traffic Engineering for Intra-domain Quality of Service Provisioning. In *Proc. of 3rd Workshop on Quality of future Internet Services (QofIS'02)*, pages 179–193. Switzerland, October 2002.
[37] J. Boyle and V. Gill and A. Hannan and D. Cooper and D. Awduche and B. Christian and W.S. Lai. *Applicability Statement for Traffic Engineering with MPLS*. IETF, Informational RFC-3346, August 2002.
[38] D. Goderis and et al. *Service Level Specification Semantics and Parameters*. Internet draft, <draft-tequila-sls-01.txt>, work in progress, December 2001. available at: www.ist-tequila.org/sls.
[39] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True. Deriving Traffic Demands for Operational IP Networks: Methodology and Experience. *IEEE/ACM Transactions on Networking (TON)*, 9(3):265–279, June 2001.
[40] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot. Traffic Matrices Estimation: Existing Techniques and Future Directions. In *Proc. of ACM SIGCOMM'02*, Pittsburgh, USA, August 2002.
[41] K. Papagiannaki, Z.-L. Zhang N. Taft, and C. Diot. Long-Term Forecasting of Internet Backbone Traffic: Observations and Initial Models. In *Proc. of IEEE INFOCOM'03*. USA, March/April 2003.
[42] P. Triminztios, T. Baugé, G. Pavlou, P. Flegkas, and R. Egan. Quality of Service Provisioning through Traffic Engineering with Applicability to IP-based Production Networks. *Computer Communications Journal, Elsevier Science*, 26(8):845–860, May 2003.
[43] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2nd edition, 1999.
[44] Z. Wang and J. Crowcroft. Quality of Service Routing for Supporting Multimedia Applications. *IEEE J. Selected Areas in Communications (JSAC)*, 14(7):1128–1234, September 1996.
[45] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to model an internetwork. In *Proc. of IEEE INFOCOM'96*, pages 594–602. USA, March 1996.
[46] P. Flegkas, P. Trimintzios, and G. Pavlou. A Policy-based Quality of Service Management Architecture for IP DiffServ Networks. *IEEE Network Magazine*, 16(2):50–56, March/April 2002.
[47] R. Yavatkar, D. Pendarakis, and R. Guérin. *A Framework for Policy Based Admission Control*. IETF, Informational RFC-2753, January 2000.
[48] P. Flegkas, P Trimintzios, G. Pavlou, I. Andrikopoulos, and C. Cavalcanti. On Policy-based Extensible Hierarchical Network Management in QoS-enabled IP Networks. In *Proc. of 1st IEEE Workshop on Policies for Distributed Systems and Networks (Policy '01)*, pages 230–246. U.K., January 2001.
[49] J. Strassner, B. Moore, R. Moats, and E. Ellesson. *Policy Core LDAP Schema*. IETF Internet draft, <draft-ietf-policy-core-schema-16.txt>, work in progress, October 2002.
[50] B. Moore, E. Ellesson, J. Strassner, and A. Westerinen. *Policy Core Information Model – Version 1 Specification*. IETF, Standards Track RFC-3060, February 2001.
[51] Y. Rekhter and T. Li. A border gateway protocol 4 (bgp-4). Internet draft, draft-ietf-idr-bgp4-17.txt, work in progress, May 2002.
[52] J. Stewart. *BGP4 : interdomain routing in the Internet*. Addison Wesley, 1999.
[53] L. Subramanian, S. Agarwal, J. Rexford, and R. Katz. Characterizing the internet hierarchy from multiple vantage points. In *INFOCOM 2002*, June 2002.
[54] S. Bartholomew. The art of peering. *BT Technology Journal*, 18(3), July 2000.
[55] Cisco. NetFlow services and applications. White paper, available from `http://www.cisco.com/warp/public/732/netflow`, 1999.

[56] W. Fang and L. Peterson. Inter-AS traffic patterns and their implications. In *IEEE Global Internet Symposium*, December 1999.
[57] P. Pan, E. Hahne, and H. Schulzrinne. BGRP: A Tree-Based Aggregation Protocol for Inter-domain Reservations. *Journal of Communications and Networks*, 2(2), June 2000.
[58] G. Huston. Analyzing the Internet's BGP routing table. *Internet Protocol Journal*, 4(1), 2001.
[59] L. Kleinrock and W. Naylor. On measured behavior of the ARPA network. In *AFIS Proceedings, 1974 National Computer Conference*, volume 43, pages 767–780. John Wiley & Sons, 1974.
[60] K. Claffy, H. Braun, and G. Polyzos. Traffic characteristics of the T1 NSFNET backbone. In *INFOCOM93*, 1993.
[61] P. McManus. A passive system for server selection within mirrored resource environments using as path length heuristics. Available from `http://www.gweep.net/~mcmanus/proximate.pdf`, April 1999.
[62] D. Awduche, A. Elmalid, I. Widjaja, and X. Xiao. A framework for Internet traffic engineering. Internet draft, draft-ietf-tewg-framework-05.txt, work in progress, May 2001.
[63] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O.Bonaventure. Interdomain traffic engineering with bgp. *IEEE Communications Magazine*, May 2003.
[64] University of Oregon. Route-views. Available from `http://antc.uoregon.edu/route-views`.
[65] A. Broido, E. Nemeth, and K. Claffy. Internet expansion, refinement and churn. *European Transactions on Telecommunications*, January 2002.
[66] K. Ishiguro. Gnu zebra 0.92a. Available from `http://www.zebra.org`.
[67] S. Bellovin, R. Bush, T. Griffin, and J. Rexford. Slowing routing table growth by filtering based on address allocation policies. preprint available from `http://www.research.att.com/~jrex`, June 2001.
[68] S. Uhlig, O. Bonaventure, and B. Quoitin. Interdomain Traffic Engineering with minimal BGP Configurations. In *Proc. of the* 18^{th} *International Teletraffic Congress, Berlin*, September 2003.
[69] S. Borthick. Will route control change the internet ? *Business Communications Review*, September 2002.
[70] B. Fortz, J. Rexford, and M. Thorup. Traffic engineering with traditional IP routing protocols. *IEEE Communications Magazine*, October 2002.
[71] T. Griffin and G. Wilfong. Analysis of the MED oscillation problem in BGP. In *ICNP2002*, 2002.
[72] Hung-Ying Tyan. *Design, Realization and Evaluation of a component-based compositional software architecture for network simulation.* PhD thesis, The Ohio State University, 2002.
[73] B. J. Premore. Ssf implementations of bgp-4. available from `http://www.cs.dartmouth.edu/~beej/bgp/`, 2001.
[74] J. H. Cowie, D. M. Nicol, and T. Ogielski. Modeling the Global Internet. *Computing in Science & Engineering*, (1):42–50, Jan/Feb 1999.
[75] T. Griffin and B. Presmore. An experimental analysis of BGP convergence time. In *ICNP 2001*, pages 53–61. IEEE Computer Society, November 2001.
[76] Z. M. Mao, R. Govindan, G. Varghese, and R. Katz. Route flap damping exacerbates internet routing convergence. In *ACM SIGCOMM'2002*, 2002.
[77] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On Power-Law Relationships of the Internet Topology. In *ACM SIGCOMM*, Sept. 1999.

[78] H. Tangmunarunkit, R. Govindan, and S. Jamin. Network Topology Generators: Degree-Based vs Structural. In *ACM SIGCOMM*, 2002.
[79] A. Medina, A.Lakhina, I. Matta, and J. Byers. BRITE: An Approach to Universal Topology Generation. In *MASCOTS 2001*, August 2001.
[80] A.L. Barábasi and R. Albert. Emergence of Scaling in Random Networks. *Sciences*, (286):509–512, October 1999.
[81] R. Albert and A. Barabasi. Statistical mechanics of complex networks. *Review of Modern Physics*, pages 47–97, January 2002.
[82] C. Demichelis and P. Chimento. IP Packet Delay Variation Metric for IPPM. Internet Engineering Task Force, RFC3393, August 2002.
[83] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. Internet Engineering Task Force, RFC 2475, December 1998.
[84] L. Xiao, K. Lui, J. Wang, and K. Nahrstedt. QoS extensions to BGP. In *ICNP 2002*, Paris, France, November 2002.
[85] G. Cristallo and C. Jacquenet. Providing Quality of Service Indication by the BGP-4 Protocol: the QOS_NLRI Attribute. Internet draft, draft-jacquenet-qos-nlri-04, Work in progress, September 2002.
[86] D. Walton, D. Cook, A. Retana, and J. Scudder. Advertisement of multiple paths in BGP. Internet draft, draft-walton-bgp-add-paths-01.txt, work in progress, November 2002.
[87] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. Internet Engineering Task Force, RFC 2460, December 1998.
[88] R. Chandra and J. Scudder. Capabilities Advertisement with BGP-4. Internet Engineering Task Force, RFC 2842, May 2000.
[89] C. Labovitz, A. Ahuja, A. Bose, and J. Jahanian. Delayed Internet Routing Convergence. August 2000.
[90] T. Li Y. Rekhter. An Architecture for IPv6 Unicast Address Allocation. Internet Engineering Task Force, RFC 1887, December 1995.
[91] W. Simpson T. Narten, E. Nodmark. Neighbor Discovery for IP Version 6 (IPv6). Internet Engineering Task Force, RFC2461, December 1998.
[92] M. Crawford. Router Renumbering for IPv6. Internet Engineeing Task Force, RFC2894, August 2000.
[93] Y. Rekhter T. Bates. Scalable Support for Multi-homed Multi-provider Connectivity. Internet Engineering Task Force, RFC2260, January 1998.
[94] J. Hagino. IPv6 multihoming support at site exit routers. Internet Engineering Task Force, RFC3178, April 2001.
[95] F. Teraoka, M. Ishiyama, M. Kunishi, and A. Shionozaki. LIN6: A Solution to Mobility and Multi-Homing in IPv6. Internet draft, draft-teraoka-ipng-lin6-02. Work in progress, November 2001.
[96] P. Tattam. Preserving active TCP sessions on multihomed. IPng Working Group Meeting Minutes, Tokio. Work in progress, September 1999.
[97] R. Stewart and al. Stream Control Transmission Protocol. Internet Engineering Task Force, RFC2960, October 2000.
[98] M. Py. Multi Homing Translation Protocol. Internet draft, draft-py-multi6-mhtp-02. Work in progress, 2001.

Mobile Networking

Chris Blondia (Ed.)[1], Nik Van den Wijngaert[1], Gert Willems[1], Olga Casals (Ed.)[2], Llorenc Cerda[2], Marcelo Bagnulo[3], Ignacio Soto[3]

[1] University of Antwerpen, Department of Mathematics and Computer Science, Universiteitsplein 1 B-2610 Antwerpen, Belgium
{chris.blondia, nik.wijngaert, gert.willems}@ua.ac.be
http://win-www.ruca.ac.be/u/pats/
[2] Technical University of Catalonia, Department of Computer Architecture, Jordi Girona, 1-3, Modul D6 E-08034 Barcelona, Spain
{olga, cerda}@ac.upc.es
http://www.ac.upc.es/recerca/
[3] Universidad Carlos III, Departamento de Ingeniera Telematica Madrid, Spain

Abstract. We point out the different performance problems that need to be addressed when considering mobility in IP networks. We also define the reference architecture and present a framework to classify the different solutions for mobility management in IP networks. The performance of the major candidate micro-mobility solutions is evaluated for both real-time (UDP) and data (TCP) traffic through simulation and by means of an analytical model. Using these models we compare the performance of different mobility management schemes for different data and real-time services and the network resources that are needed for it. We point out the problems of TCP in wireless environments and review some proposed enhancements to TCP that aim at improving TCP performance. We make a detailed study of how some of micro-mobility protocols namely Cellular IP, Hawaii and Hierarchical Mobile IP affect the behavior of TCP and their interaction with the MAC layer. We investigate the impact of handoffs on TCP by means of simulation traces that show the evolution of segments and acknowledgments during handoffs.

1. Introduction

Mobility support in IP networks is in the final steps of the standardization process within the IETF. This specification defines basic tools needed to cope with mobile nodes. However, it is well known in the research community that multiple ulterior optimizations are required in order to provide a service comparable with those provided in a non-mobile environment. In this chapter, a number of critical issues with respect to mobility are addressed.

The state-less auto-configuration procedure is not optimized for real-time services, since obtaining a Care-of-Address requires a Duplicate Address Detection, which may take a default value of one second. This latency is unacceptable in an environment for interactive voice communications. Section 2 presents a possible modification of the

M. Smirnov et al. (Eds.): COST 263 Final Report, LNCS 2856, pp. 180-206, 2003.

Duplicate Address Detection mechanism of IPv6 with the aim of reducing the time needed to perform a handover.
In section 3, protocols are investigated that aim at realizing seamless handover, i.e. handovers without packet loss and without introducing extra packet delay and delay jitter. Their performance is studied and the influence of several system parameters is investigated.
The TCP transport protocol used in the Internet is a variable window protocol where losses are considered as congestion signals. This may cause TCP to behave poorly in wireless networks, which are characterized by losses due to transmission errors and handoffs. In section 4 we investigate the impact of transmission errors on TCP and the causes of packet losses during the handoffs.

2. Optimizations for Mobility Support in IPv6

2.1. Introduction

IPv6 introduces enhanced facilities for mobility support, among which we can find State-less Auto-configuration mechanism. However, general auto-configuration procedure is not optimized for mobile nodes as it will presented next. Since in order to take full advantage of multimedia capabilities of current mobile devices, the network infrastructure must provide an uninterrupted flow of information to appropriately support real time traffic. However, the requirement for performing Duplicate Address Detection (DAD) in the address autoconfiguration mechanism limits the performance of mobility in IPv6, provided by Mobile IPv6 [1]. Using this protocol, a Mobile Node (MN) that joins a subnet must configure an on-link address in that subnet, the Care-of-Address (CoA), before being able to communicate. According to the Stateless Address Autoconfiguration mechanism presented in [2], before using the CoA the MN must perform DAD for that address in order to guarantee its uniqueness on the link. It should be noted that DAD is a time consuming process. Although this it is not an issue for a desktop computer that is booting, the time required for DAD is critical in a mobile environment, since during this time the MN can not communicate and besides, all active communications of the MN are interrupted. The time required to perform DAD has a default value of one second [2], [3], a value subjectively deemed as not acceptable for interactive voice communications [4]. The Mobile IPv6 (MIPv6) specification [1] identifies the aforementioned problem and states that a MN can decide not to perform DAD, pointing this as a trade-off between safety and the time needed for the DAD procedure.

In this section, the usage of random numbers to create the Interface Identifier part of the IPv6 addresses is explored, and the risk of using these addresses without previously performing DAD is assessed. It should be noted that this solution is not restricted to a particular data-link layer technology, although it can be optimized in particular cases, such as GPRS, in which collision can be avoided by the GGSN (Gateway GPRS Support Node).

2.2. Risk Assessment

In this sub-section we will asses the risk of using randomly generated Interface Identifiers (IIDs) in IPv6 aggregatable addresses [5] without previously performing DAD. In order to do that, we will quantify the probability of a duplicate address event in several relevant scenarios and we will compare it with the probability of other critical events.

2.2.1. Duplicate Address Event Probability Calculation and Bounding

In the considered hypothesis, the Interface Identifier part of the IPv6 address is generated randomly, meaning that the node will use a 64 bit long random number as the IID. Actually, only 62 bits of the IID will be generated randomly, since the IID´s semantics defined in [6] imposes that the u bit must be set to "local" and the g bit must be set to "individual".

Considering that n is the number of possible IIDs (i.e. $n = 2^{62}$) and k is the number of interfaces (i.e. mobile nodes) on the same link, we will now calculate the probability of collision of two or more randomly generated IIDs:
We will represent the k IIDs in a link as a sequence of 62 bit long random variables I_i:

$I_1, I_2, ..., I_k$ sequence of random integer variables with uniform distribution between 1 and $n \left(k \le n\right)$

We would like to obtain the probability that two or more I_is collide, i.e. $I_i = I_j$

This is well known mathematical problem, called the "birthday problem", whose classical formulation is as follows: we want to calculate the probability that in a group of k people, al least two of them have the same birthday.

We model the birthday as a integer random variable, with uniform distribution between 1 and n (in this particular case n is the number of possible birthdays i.e. n=365)

Then, the number N of ways that we can choose k values out of n with no duplicates would be:

$$N = n \cdot (n-1) \cdot ... \cdot (n-k+1)$$

On the other hand, the number of possibilities for choosing k elements out of n, without the restriction of not having any duplicates is n^k
Then, the probability of not having a collision when we select k elements out of n is:

$$P_{NO}(n,k) = \frac{n!}{(n-k)! \cdot n^k}$$

So, the resulting expression for the probability of the collision of one or more I_i is:

$$P(n,k) = 1 - \frac{n!}{(n-k)! \, n^k} \quad \textbf{(1)}$$

In our particular case, $n = 2^{62}$, and k may vary depending on the considered scenario. We will now obtain an upper bound to $P(n,k)$ in order to simplify calculations (especially to avoid $n!$ computation)

Performing simple computations in equation (1), we easily obtain:

$$P(n,k) = 1 - \left\{ 1 \cdot \left(1 - \frac{1}{n}\right) \cdot \left(1 - \frac{2}{n}\right) \ldots \left(1 - \frac{k-1}{n}\right) \right\} \quad \textbf{(2)}$$

Since

$\forall i \in [1,2,\ldots,k-1] \Rightarrow \frac{i}{n} \leq \frac{k-1}{n}$,

and considering that $k < n$, then

$$1 - \left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right) \ldots \left(1 - \frac{k-1}{n}\right) \leq 1 - \left(1 - \frac{k-1}{n}\right)^{k-1}$$

Applying this last result to equation 2, we can obtain the following bound B:

$$P(n,k) \leq \frac{n^{k-1} - (n-k+1)^{k-1}}{n^{k-1}} = B \quad \textbf{(3)}$$

We will next perform some calculations in order to quantify the order of magnitude of the probabilities involved:

We will bound $P(n,k)$ for the following values of *k*, which we consider to be representative of usual situations

$P(2^{62},20) \leq 7.8e-17$

$P(2^{62},100) \leq 2.1e-15$

$P(2^{62},500) \leq 5.4e-14$

$P(2^{62},1000) \leq 2.2e-13$

$P(2^{62},5000) \leq 5.4e-12$

In order to fully seize the magnitude of the probabilities stated above, we can compare them with the probabilities of some rare events. For instance, according to Table 1.1 in [7], the probability of being killed by a lightning (per day) is about 1.1 e-10. Then, a mobile phone user should be more worried about being killed by a lightning in a given day than to have an interface identifier repeated when he performs a handoff.

Another relevant parameter that can be considered when evaluating the above probability, is the probability of a failure in a network device, since this failure would have similar effects i.e. the user can not continue the communication. So, the probability that a network device were not working properly in a given moment (when the user joins the network, for instance) can be calculated as follows:

$$P_{NEFails} = \frac{MTTR}{MTBF + MTTR}$$

Being MTTR the Meat Time To Repair and MTBF the Mean Time Between Failures.

Good network equipment can have an MTBF of 300,000 hours and if we suppose that some backup device is available, the MTTR stands for the time needed to replace the faulty element, e.g. 0.1 hour (6 minutes). In this case, $P_{NEFails} = 3.3e-7$.

We can see that $P_{NEFails}$ is several orders of magnitude higher than $P(n,k)$ in the cases calculated above.

2.2.2. Scenarios

We have quantified the probability of a collision of two or more IIDs. However, this probability is not the most relevant parameter when we try to evaluate and compare the probability of failure of the system, since a mobile user will join multiple links in a given period. So, it is relevant to quantify the probability of at least one collision when a user performs multiple handoffs.

As we stated above, $P(n,k)$ is the probability of a collision of two or more IIDs when there are *k* interfaces in the same link. Hence, it can be derived that the probability of having at least one collision after joining *m* links is:

$$P(n,k,m) = 1-(1-P(n,k))^m \quad \textbf{(4)}$$

According to the bound *B* presented in equation 3 and considering that both $P(n,k)$ and *B* are lower than 1, we can infer the following bound:

$$P(n,k,m) \le 1-(1-B)^m \quad \textbf{(5)}$$

Therefore, in order to estimate the probability of a collision event during a given period, for instance a year, we must first establish a reasonable number of handoffs per day. If we consider 140 handoffs per day, which seems to be a considerable number, this would mean about 50.000 handoffs per year, i.e. *m*=50.000. Then the probability of having at least one collision over a year during which the mobile node has joined 140 links of 500 nodes per day is:

$$P(2^{62},500,50.000) \le 2.7e-9 \quad \textbf{(6)}$$

And, if the considered links have 5.000 nodes each, instead of 500, the probability is:

$$P(2^{62},5.000,50.000) \le 2.7e-7 \quad \textbf{(7)}$$

Considering that each time there is a collision there are two users affected, and not taking into account the collision of 3 or more IIDs for this estimation, there will be 6 users out of 1.000.000.000 that will have a communication problem during this year, if users make 140 handovers per day in networks containing 500 interfaces. In the case that users make 140 handovers per day in networks containing 5.000 interfaces, there will be 6 users out of 10.000.000 that will have a communication problem during this year. This probability could be contrasted with some network availability

data provided by, for instance, mobile operators, but this data has proven to be extremely difficult to find.

2.3. Implementation Issues

In this section, we will address some implementation issues regarding random IIDs generation and related security concerns.

2.3.1. Random Numbers Generation

When considering the usage of random IIDs, the random number generation process must be properly addressed since it is essential to guarantee the statistic uniqueness of the identifier. Several methods have been proposed [8] to generate pseudo-random numbers based on information available in most platforms, such as laptops. However, in some cases, such as mobile phones, the resources required to perform appropriate random number generation may not be available. In such cases, it should be noted that it is not necessary to create the identifier in real time, as long as randomness were guaranteed. This means that when the node joins the network the identifier could have been created already in advance to a network change. It could even be pre-configured in the interface driver with the node using the same identifier without changing the probabilities calculations stated above; this is analogous to the day of birth in the birthday problem. This would reduce the complexity in the nodes, although a mechanism should be provided in order to solve recurrent collisions, caused for example, when two habitual users of the same segment collide.

2.3.2. Security Concerns

Randomly generated IIDs have also been considered in order to improve security. In particular, its usage has been proposed to ensure anonymity [9] and even to carry cryptographic information when Cryptographically Generated Addresses are used [10]. These proposals are fully compatible with the solution of this document so they can get the benefit of better performance by avoiding DAD.

3. Mobility Management in IP Networks

3.1. Introduction

In this Section, we investigate methods to solve the problem of host mobility on layer 3. The basic problem is how to route packets to a Mobile Node (MN) when it moves from one point of attachment to another. The dual character of an IP address plays a central role: on the one hand the IP address uniquely identifies a MN, and on the other hand it identifies the location of the MN.
In what follows, we discuss several solutions to this host mobility problem. These solutions should ensure (as much as possible) a seamless handover, i.e. a handover without packet loss and without introducing extra packet delay or delay jitter. In

addition, the messages required to implement these solutions and their storage and processing needs in routers should be minimal. Moreover, these mechanisms should have a maximal compatibility with other Internet protocols, in particular with QoS provisioning schemes. Scalability is another important requirement that has to be fulfilled.

3.2. IP Mobility Protocols

In this subsection, we give an overview of important IP mobility protocols. We start with the standard IETF solution, namely Mobile IP. Next we discuss two important classes of micro-mobility mechanisms, namely hierarchical tunnelling mobility protocols and host-specific routing protocols. Finally we discuss some low latency handoff schemes.

3.2.1. Mobile IP

Mobile IP (MIP) is the standard IP solution for host mobility. MIP uses two addresses, a home address (HoA) acting as a permanent Layer 3 identifier and the Care-of-Address (CoA) acting as a temporary routable address to indicate the actual location of the MN.
An agent located in the MN's home network, called the Home Agent (HA) maintains the binding <HoA, CoA>. The HA intercepts packets destined for the MN and tunnels them to the MN's CoA. When the MN changes subnet (and hence CoA), it needs to registrer the new CoA with the HA (i.e. a binding update). This registration may take a considerable time which, together with the delays introduced by the establishment of the new tunnels, may cause unacceptable packet loss and the introduction of an important signalling overhead. For the basic operation of MIP, we refer to [13] and [1].
Due to its robustness and simplicity, MIP seems to be a suitable protocol for handling global or macro-mobility (e.g. mobility between domains owned by different operators). Local (and hence more frequent) handovers need to be handled by more optimized, so-called micro-mobility protocols. In what follows, we consider two classes of micro-mobility protocols, namely schemes based on hierarchical tunnelling and protocols based on host-specific routing.

3.2.2. Hierarchical Tunneling Mobility Protocols

Hierarchical tunneling schemes, such as MIP Regional Registration (IPv4) ([14]) or Hierarchical Mobile IP (MIPv6) ([15]), handle local movements by maintaining the MN's location information in a distributed form by a hierarchy of Foreign Agents (FA) organized in a tree structure. Movements of the MN between different points of attachment only involve binding updates at the optimal point in the tree. In such a system, the HA tunnels the traffic to the root of the tree and each intermediate FA on the route to the MN decapsulates and re-encapsulates the packets as they are forwarded down the tree towards the MN's actual position.

3.2.3. Host-Specific Routing Protocols

This class of micro-mobility schemes avoids the overhead introduced by the tunnelling schemes by using host-specific routes towards the current location of the MN. These host-specific routes are created and maintained by explicit signalling or by snooping data packets. We discuss two important examples, namely Cellular IP (CIP) ([16]) and Handover Aware Wireless Access Internet Infrastructure (HAWAII) ([17]).

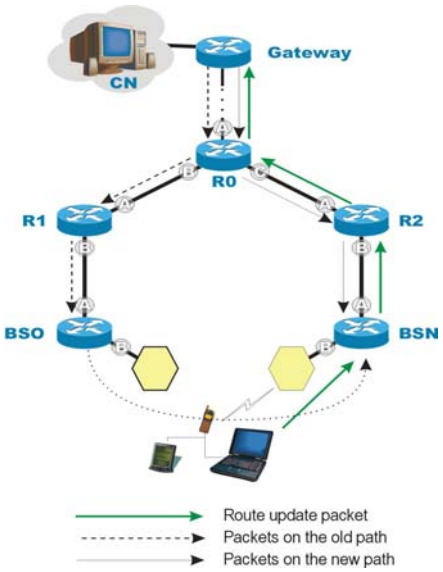


Figure 1: Cellular IP

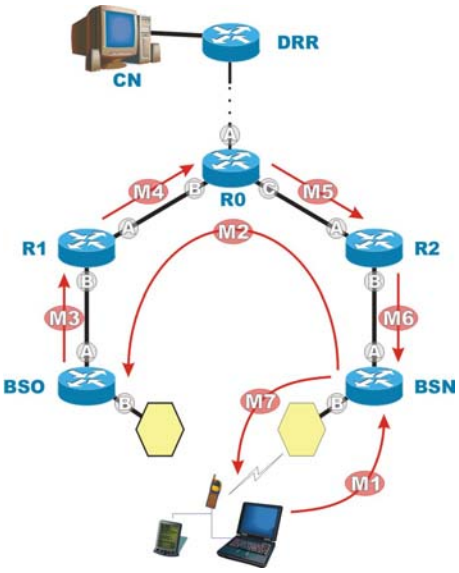


Figure 2: HAWAII

Cellular IP

A Cellular IP ([16], [18], [19]) access network is connected to the Internet via a gateway router. A MN attached to an access network will use the IP address of the gateway as its Mobile IP CoA.

After power up a MN has to inform the gateway router about its current point of attachment by means of a route update message. This message is received by the base station and forwarded to the gateway on a hop-by-hop basis. Each Cellular IP node maintains a route cache in which it stores host based routing entries. The path taken by uplink packets is cached by all intermediate nodes. To route downlink packets addressed to the MN, the path used by recently transmitted packets from the MN and which have been cached is reversed.

As the routing entries in the nodes are soft state, a MN that has no data to transmit, has to send periodically special IP packets to maintain its routing path. Paging is used to route packets to MNs that have not been transmitting or receiving data for a while

and whose routing cache entries have timed out. Paging caches are not maintained in each node and have a longer timeout value. If a node without a paging cache, receives a packet for a MH for which it has no routing cache entry, it will broadcast it to all its downlink neighbours.
In Cellular IP a mobile host initiates a handoff (based on signal strength measurements) by sending a route update packet to the new base station (see Figure 1). This packet travels in a hop-by-hop manner from the base station to the gateway router and reconfigures the route cache entries in the Cellular IP nodes along its way. Cellular IP supports different types of handoff schemes. In the hard handoff scheme, the wireless interface of the MN switches from one BS to another at once. In the semi-soft handoff scheme, the MN is assumed to be able to transmit a route update packet to the new base station while still listening to the old base station. By duplicating packets, the packet loss observed in hard handoff may be reduced.

HAWAII
HAWAII ([17]) creates host-specific routing entries in the routers by explicit signaling messages triggered by the MN. Four different path set-up schemes have been defined which are classified into two different classes: two forwarding schemes, namely Multiple Stream Forwarding (MSF) and Single Stream Forwarding (SSF) and two non-forwarding schemes, namely Unicast Non-Forwarding (UNF) and Multicast Non-Forwarding (MNF). We limit the description and the analysis (see Section 2.3.1.) to the MSF scheme.
The MSF schemes is described by means of the following messages (see Figure 2). At the instant of handoff, the old base station, BSO, loses contact with the MN and at the same time the MN sends a MIP registration message (M1) to new base station, denoted by BSN. The latter sends a path setup update message M2 to the BSO. When M2 arrives at BSO, the BSO starts to forward all packets with destination MN via router R1, including those packets that arrive after the handoff instant and that were stored in a forwarding buffer at the BSO. For that purpose, BSO adds a forwarding entry to its routing table indicating that packets for MN should leave the BSO via interface A. BSO sends the path setup message (M3) to R1, who adds a forwarding entry to its routing table indicating that packets for the MN should leave the R1 via interface A. R1 sends the path setup message (M4) to R0, who adds a forwarding entry indicating that packets for the MH should leave the R0 via interface C. From this instant on, all packets arriving at router R0 are sent directly to BSN. The path setup message continues (M5 and M6) triggering similar actions until it reaches BSN.

3.2.4. Low Latency Handoff Schemes

In this paragraph we discuss three different handoff schemes that aim at low latency handoffs: smooth handoff, pre-registration and post-registration.

Optimized Smooth Handoff
Optimized smooth handoff was proposed in [20]. We make the assumptions that the route optimization problem (i.e. the triangle routing problem) is assumed solved adequately. Furthermore, in what follows, we assume a network with a hierarchical Foreign Agent (FA) architecture. As soon as the MN has obtained its new regional CoA, it will register this address with its GFA. This is achieved by sending a

Registration Request Message to the new FA, who on his turn sends a registration message to the GFA. As part of this registration procedure, the MN may add to the registration request message a Previous Foreign Agent Notification extension. The new FA will then send a Binding Update Message to the previous FA, with the request to reply with an acknowledgement. This acknowledgment is tunnelled to the new FA, who should forward it to the MN (this may involve unauthorized traffic to the MN). In this way the MN is notified that the previous FA has the new CoA of the MN. This binding update the previous FA receives from the new FA, allows packet forwarding mechanism. When a packet arrives in the previous FA, the binding cache is checked and the packet is tunnelled to the new FA, who delivers it to the MN. However, packets arriving at the previous FA after the MN left and before the binding update message from the new FA is received are lost. In order to avoid this packet loss, FAs are provided with a circular buffer referred to as the Forwarding Buffer. When a tunnelled packet arrives at the previous FA, it is decapsulated, delivered to the MN (if possible) and copied into a buffer. When the previous FA receives a binding update originating from a previous foreign agent notification, these buffered packets are re-tunnelled to the new FA and all packets arriving at the previous FA with destination the MN are immediately tunnelled to the new FA. In order to avoid duplicate packets, the MN includes the pair of source address and datagram identification of the most recent received packets in the registration request that is sent to the previous FA, who uses this pair to drop the buffered packets that have been received by the MH.

Pre-registration

The Pre-Registration handoff scheme ([21]) is based on an anticipated Layer 3 handoff by means of Layer 2 triggers. The L2 trigger contains the new FA's IP address and this allows the Mobile Node (MN) to communicate via the previous FA with the new FA while still being connected with the previous FA. The MN sends a registration request to the new FA via the previous FA (if the L2 handoff is not completed) and the new FA issues a regional registration to the Gateway FA (GFA). The latter sends a registration reply message to the new FA. Until the MN actually completes the L2 handoff to the new FA and establishes the new L2 link, the new FA can receive packets for which it does not have a link layer connection. These packets may be buffered in the new FA.

Post-registration

In the Post-Registration scheme ([21]), the registration occurs after the L2 handoff is complete. The L2 trigger initiates the set-up of a bi-directional edge tunnel (BET) between the previous FA and the new FA. When the previous FA receives the L2 Line Down (LD) trigger, it starts forwarding packets destined to the MN via the BET to the new FA. When the new FA receives an L2 Line Up (LU) trigger, it delivers the packets received from the previous FA to the MN. Eventually the MN performs a formal MIP registration.

3.3. A Generic Analytical Model for IP Mobility Protocols

We propose an analytical model to compute the packet loss and the delay experienced by a UDP stream (i.e. a constant bit rate packet stream) originating from a Corresponding Node (CN) with destination the MN, when the MN switches from one access point to another. For computational tractability reasons, all routers are modelled as simple M/M/1 queues. From a performance modelling point of view, the packets can be subdivided into different classes. The time intervals that determine to which class a packet belongs are obtained from the end-to-end delay a message experiences when travelling form one node in the network to another node. Since all routers are modelled as M/M/1 queues, the length of these time intervals is the sum of exponentially distributed random variables and constants, and therefore computable in a fairly straightforward way. While travelling to the MN, each packet follows a specific path of routers according to the class it belongs to. Again due to the M/M/1 assumption, this path is the sum of exponentially distributed random variables and constants, and therefore the end-to-end delay can be computed easily. The model also allows the computation of the buffer requirements and possible packet loss probabilities. The models for the handoff schemes discussed in the previous section are found in the following papers: Cellular IP [22], HAWAII [23], Smooth Handoff [24], Pre-Registration [25] and Post-Registration [26].

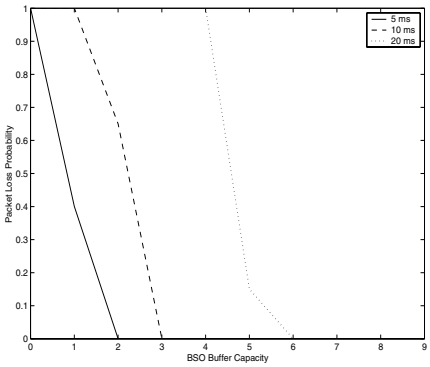


Figure 3: Packet loss probability in the Forwarding Buffer

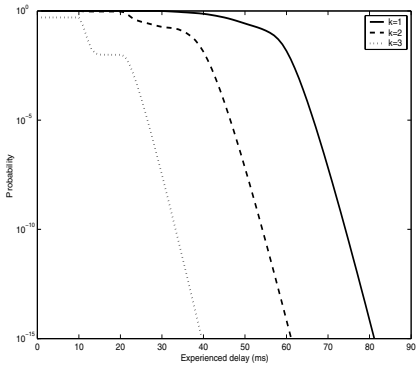


Figure 4: Delay experienced by packets due to forwarding

3.4. Performance Analysis of IP Mobility Protocols

In this Section we apply the generic analytical model for IP Mobility protocols described in the previous Section to HAWAII, Cellular IP, Smooth Handoff, Pre- and Post-Registration Handoff. A comparison of these mobility protocols can be found in [27].

3.4.1. HAWAII

Consider the system depicted in Figure 2, where each router (including the Base Stations) has the following characteristics: the service rate μ equals 10 packets per ms, the load is given by ρ = 0.8, and the propagation delay to each neighboring router is variable (5 ms, 10 ms and 20 ms). We consider a stream of packets arriving at Router R0 with a constant packet interarrival time of *T*=20 ms.
For this system, Figure 3 shows the packet loss probability as a function of the capacity of the forwarding buffer at the Old BS for the different propagation delays between neighboring routers (5 ms, 10 ms and 20 ms). Next, we compute the delay due to forwarding for this system. Figure 4 shows the probability that the *k*-th packet after handoff experiences a delay of at least *t* ms due to the forwarding scheme.

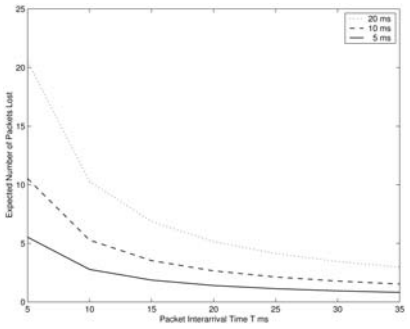


Figure 5: Hard handoff : Expected number of packets lost in BSO

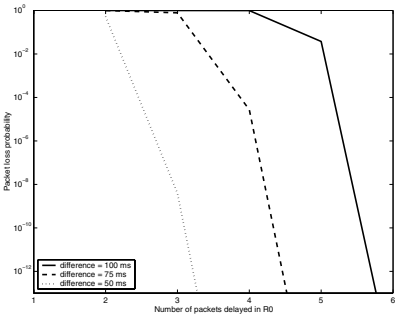


Figure 6: Packet loss probability as a function of number of delayed packets for different values of difference in length of new and old path

3.4.2. Cellular IP

First we evaluate the case of the Hard Handoff scheme. Consider the reference network as described in 2.3.1. We consider a stream of packets arriving at Router R0 with a constant packet interarrival time, taking values between 5 ms and 35 ms. Figure 5 shows the expected number of packets that are lost at the BSO due to the hard handoff, for variable packet interarrival times and different values of propagation delay. Clearly the expected number of lost packets increases when the propagation delay increases and the interarrival time decreases.
Next, for the Semi-soft Handoff scheme we compute the packet loss probability as a function of the number of packets that are delayed in the R0 buffer before the first packet is released, in order to cope with the different propagation delay between R0-BSO and R0-BSN. We assume this difference to be 50 ms, 75 ms and 100 ms. The three curves in Figure 6 show the loss probability of packets due to early arrival in BSN for different values of the buffer capacity in R0.

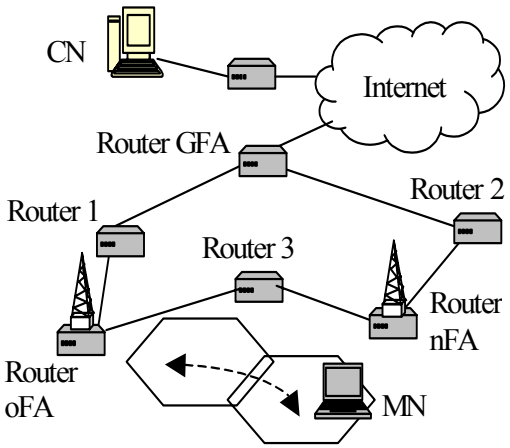


Figure 7: Reference Network for smooth handoff

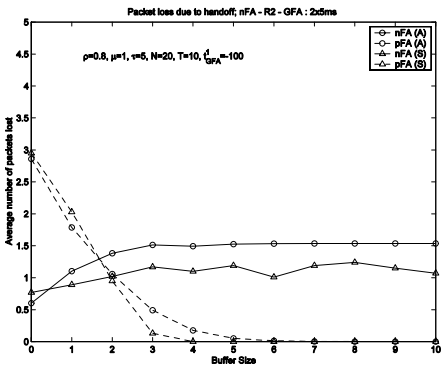


Figure 8: Packet loss as function of the buffer size

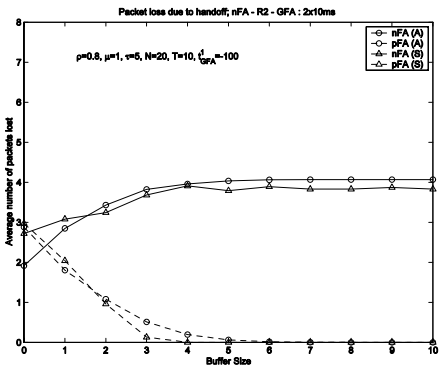


Figure 9: Packet loss as function of the buffer size

3.4.3. Optimized Smooth Handoff

Consider the network depicted in Figure 7 with the following system parameters. Each router is loaded up to 0.8, the propagation delay between routers is $\tau = 5$ ms, the average processing time of a packet in a router is 1ms.

In Figure 8 and Figure 9 the expected number of lost packets is shown as a function of the buffer size at the previous FA. The analytical results (A) are compared to simulation results (S). The expected packet loss due to buffer overflow is given by the dashed line, while the solid line represents the additional loss at the new FA, due to early arrival. Figure 8 shows the results for link delays equal to 5ms on every link, while for Figure 9, the 2 links on the nFA-GFA path are increased to 10ms each.

Obviously, the loss in the buffer at the previous FA diminishes when the buffer size is increased. The packets that are lost in case of very small buffer size do go through the buffer when this buffer size increases. But in the latter case they possibly contribute to the number of lost packets at the new FA, hence the rise of the solid curve for small buffer sizes. This is especially true for the case of 10ms link delay on the nFA-GFA path, because then the whole buffer is likely to arrive too early at the new FA (i.e. before the registration reply message from the GFA has arrived). In other words, in the case of long delay on the nFA-GFA path, if a packet is not dropped at the previous FA, it will most likely be lost at the new FA.

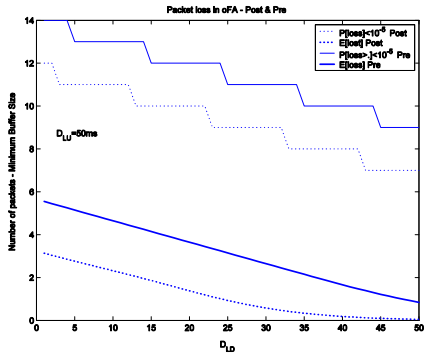


Figure 10: Packet loss in oFA

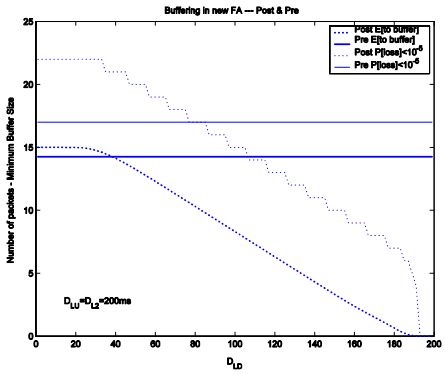


Figure 11: Buffers required at nFA

In order to avoid packet loss at the previous FA, the forwarding buffer need to be dimensioned such that it can store packets of the order of the product bit rate of the stream times delay (MN - new FA – previous FA). The loss at the new FA on the other hand depends on the difference between the distance (new FA – GFA) and (new FA – previous FA). If the latter is smaller than the former, then packets may get lost. A possible solution would be to provide the new FA with a buffer to store temporarily unauthorized traffic until the registration reply from the GFA arrives at the new FA. Another solution consists of sending the binding update message from the new FA to the previous FA via the GFA in order to allow the registration reply message to arrive before the first forwarded packet. A similar solution has been applied in the Multiple Stream Forwarding scheme of the HAWAII.

3.4.4. Pre- and Post Registration

Consider the network depicted in Figure 7. Let τ_1 represent the propagation delay on the links connecting the Gateway and the oFA and also on the links connecting the Gateway and the nFA, and let τ_2 represent the propagation delay on the links connecting the oFA and the nFA. In order to compare the two schemes, we have to relate the respective triggers. We make the following assumptions. We let $t_0 = 0$ be the start of the handoff and we assume that $D_{L2} = D_{LU}$ (i.e. the handoff is completed when the nFA receives the LU trigger).

In a first example, we consider a system with the following parameters. A CN transmits 500 byte packets every T=10 ms in a network with μ=1, τ_1 =5 ms and τ_2 =3 ms. $D_{L2} = D_{LU} = 50ms$. Figure 10 shows the lost packets in the oFA for varying D_{LD}.

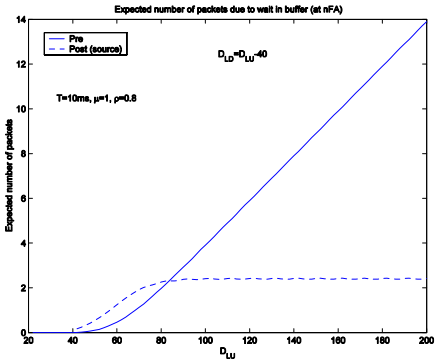


Figure 12: Buffers required in nFA

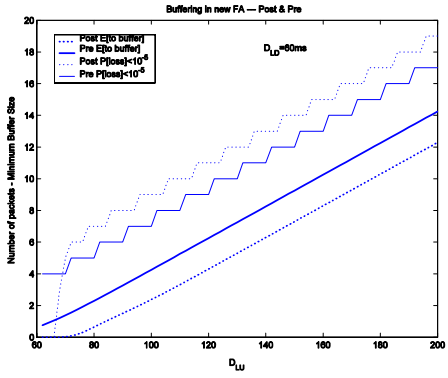


Figure 13: Buffers required in nFA

Next in Figure 11 we show the expected number of packets that need to be buffered in the nFA, in order to obtain zero packet loss (mean and 1-10^{-5} quantile). We let $D_{L2} = D_{LU} = 200ms$ and vary D_{LD}.

Whereas in the previous example $D_{L2} = D_{LU}$ was kept constant, in the next two examples we let $D_{L2} = D_{LU}$ vary. First in Figure 12, also D_{LD} varies, such that $D_{LU} - D_{LD} = 40ms$. In the last example we let $D_{LD} = 20, 40 \text{ and } 60$ ms and the corresponding results for the number of packets that need to be buffered in nFA are shown in Figure 13.

4. TCP in Wireless Networks

4.1. Introduction

The TCP transport protocol used in the Internet is a variable window protocol where losses are considered as congestion signals. Basically, TCP increases the window as acknowledgments are received and reduces the window when losses are detected. Therefore, in order TCP to perform well, it is needed that packet losses are mainly due to congestion. This may cause TCP to behave poorly in wireless networks, which are characterized by losses due to transmission errors and handoffs.

In this section we investigate the impact of wireless networks on TCP from two points of view. First, in section 3.2, we consider the TCP performance degradation

introduced by a wireless media having transmission errors. We also study the capability of several mechanisms to reduce this performance degradation. Then, in section 3.3, we investigate the packet losses due to the handoffs using some of the micromobility protocols analized in section 3.2.

4.2. TCP in a Wireless Environment

In this section we point out the problems of TCP in wireless environments and review some proposed enhancements to TCP that aim at improving TCP performance for wireless and mobile nodes.
We analyze the TCP performance degradation introduced by a wireless media having transmission errors and the capability of the following mechanisms to correct it (i) improvements over the TCP Reno release (SACK [29, 33], New-Reno [30] and FACK [34]); (ii) improvements over the classical "tail dropping" mechanism used by routers (RED [28, 31] and ECN [35]). A TCP implementation based on the usage of ECN routers that would be able to distinguish among losses due to congestion and losses due to transmission errors is analyzed in [38].

4.2.1. Simulation Framework

The simulator used to obtain the results shown in this paper is an event driven simulator written in C++. In the following the assumptions made for the simulation of the TCP module and the network topology are described.

The TCP Module
We assume a greedy TCP module (which has always segments ready to send). Our TCP module passes the maximum number of segments allowed by the TCP window to the network driver. The segments are immediately given to the driver after the TCP module initialization, when an ack allows the transmission of more segments, or when a time-out expires.
For the network driver we have assumed an infinite queue (i.e. with no losses) which stores the segments received by the TCP module until they are sent into the transmission link. However, we have considered the buffer occupancy at the TCP receiver due to segment reordering. Therefore, in the simulations the TCP receiver always advertises a window equal to the free buffer space. We have also assumed that the TCP receiver sends an ack for each received segment. A realistic simulation is done of the slow timer used by TCP for the segments retransmission control (see [37]). In fact, the slow timer function is called each time interval equal to the granularity used by the TCP module as in a real implementation.
For sake of simplicity, we have not considered queuing delays but only the propagation delay in the return path of the acks. However, in order to avoid phase effects we have added a small random component in the return path delay of the acks.
The sources run the different TCP implementations with the following parameters:
1. Segments of 576 bytes (this is the default value used by TCP if no MSS is indicated at the connection setup).
2. Buffer space at the TCP receiver equal to 50 segments.

3. The timestamp option is used for the round trip measurements (see [36]).
4. We use the retransmission time-out mechanism described in [32].

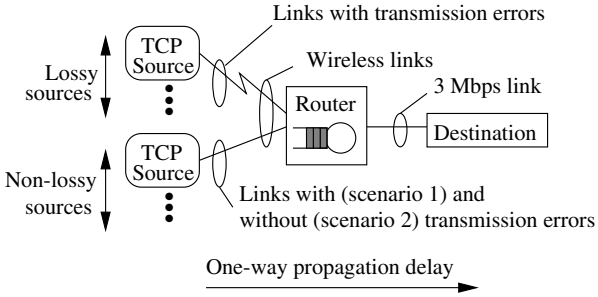


Fig. 14: Simulation scenarios.

Simulation Topology and Scenarios

Fig. 14 shows the network topology we have simulated in this paper. It consists of 20 TCP sources that compete for a congested link at the router output. The transmission links of the sources are 1 Mbps (we shall refer to these links as the wireless links), and the congested link at the router output is 3 Mbps.

We have considered two scenarios:

1. ***Lossy* scenario**: all the wireless links have the same transmission error rate,
2. ***half-lossy* scenario**: only half of the wireless links have transmission errors. We shall call *lossy* sources the sources having a transmission link with transmission errors and *non-lossy* otherwise.

1. In the simulation the errors have been introduced only in the forward direction, i.e. no acks are lost due to transmission errors. We have used a Bernoulli distribution in order to generate these transmission errors (i.e. each segment is lost due to transmission errors with probability *loss_p* and properly transmitted with.
3. probability 1 – *loss_p*). Several simulations have been carried out for each of these scenarios changing the behavior of the TCP sources and the router as follows:
4. **TCP sources**: (i) standard TCP-Reno [36], (ii) TCP with the New-Reno improvement [30], (iii) TCP with the SACK implementation [29] and (iv) TCP with the FACK implementation [34].
5. **Router**: (i) "tail dropping", (ii) with the RED discarding policy [31] and (iii) with the ECN-RED marking mechanism [35].

   Finally, simulations have been done varying the following parameters:

2. **Number of sources**: this has been set to (i) 10 sources (5 *lossy* and 5 *non-lossy* in scenario 2) and (ii) 40 sources (20 *lossy* and 20 *non-lossy* in scenario 2),
3. **end-to-end propagation delay**: this has been set to (i) 1 ms and (ii) 20 ms,
4. **TCP granularity**: this has been set to (i) 50 ms and (ii) 500 ms.

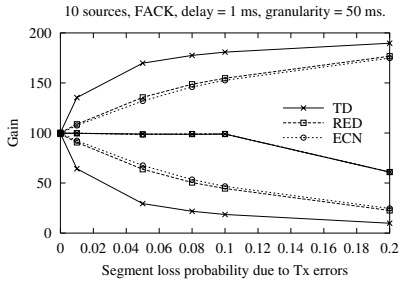


Fig. 15.A

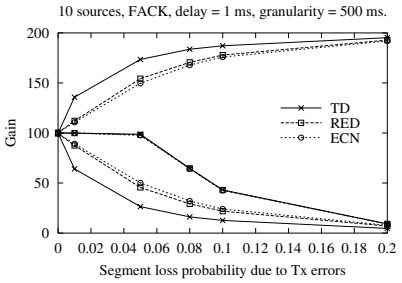


Fig. 15.B

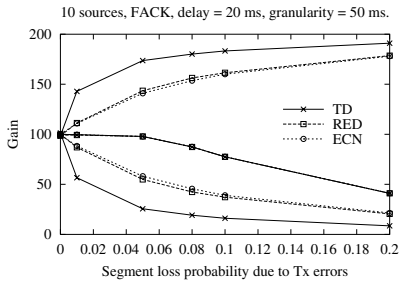


Fig. 15.C

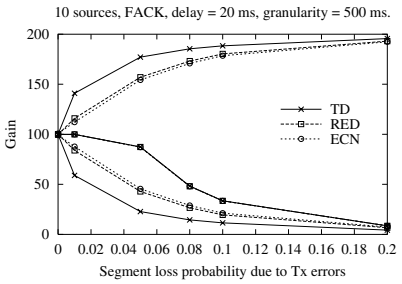


Fig. 15.D

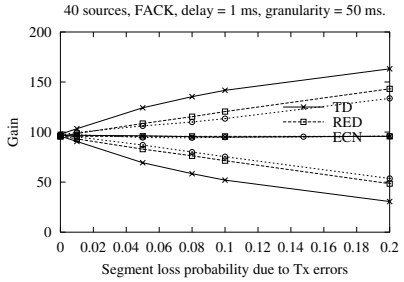


Fig. 15.E

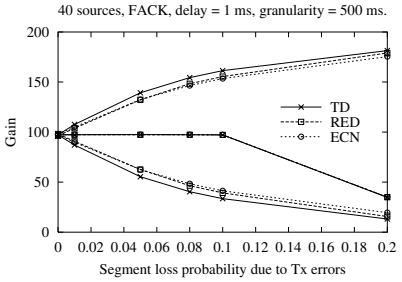


Fig. 15.F

Fig. 15: Gain obtained in the lossy and half-lossy scenarios varying the number of sources, delay and granularity using a tail dropping (TD), RED and ECN router.

The buffer size of the router has been fixed to 150 segments in all cases. RED and ECN-RED have been implemented with the following parameters (see [31]): `LowTh` = 20 segments, `HighTh` = 60 segments, `wq` = 0.5, `maxp` = 0.02

4.2.2. Numerical Results

Each of the graphs of Fig. 15 and Fig. 16 shows the results obtained for the lossy and half-lossy scenarios described in section 0 with a different set of parameters. What we call *gain* in the graphs is given by the average goodput obtained for each group of sources having the same ratio of transmission errors to the fair goodput (the link rate divided by the number of sources) multiplied by 100. We compute the goodput of one source as the sequence number increment achieved during the simulation multiplied

by the segment size in bits divided the simulation time. Remember from section 0 that in the lossy scenario all the sources have the same segment loss probability due to transmission errors and in the half-lossy scenario only half of the sources have transmission errors. This loss probability is given in the abscissa and we shall refer to it as *loss_p*. Therefore, only one point is depicted in the graph for each simulation with the lossy scenario, and two points corresponding to the group of sources having and not having transmission errors are depicted in the half-lossy scenario. Note that these points are easy to identify: (i) those in the middle of the graph correspond to the lossy scenario and (ii) those in the top and the bottom of the graphs respectively correspond to the lossy and non-lossy sources of the half-lossy scenario.

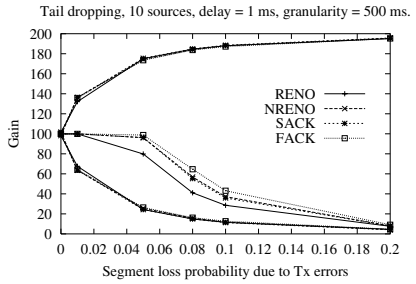


Fig. 16: Gain obtained in the lossy and half-lossy scenarios using different TCP implementations.

Note also that in Fig. 15 the curves obtained for each of the three types of routers (tail dropping, RED and ECN) have been superimposed for each scenario. The TCP implementation used for these graphs is FACK. Instead, in Fig. 15 the curves obtained for each of the TCP implementations (Reno, New-Reno, SACK and FACK) have been superimposed. In the following some general guidelines are derived from these graphs.

Unfairness: the first effect that becomes apparent from the graphs of Fig. 15 arises when comparing the results obtained with the lossy and half-lossy scenarios. In all cases the degradation of the gain obtained by the sources having losses due to transmission errors is higher when the congested link is to be shared with sources having no transmission losses. For example, Fig. 15.A shows that only for values of *loss_p* higher than 0.1 the gain starts to degrade in the lossy scenario. Instead, the gain of sources having transmission errors starts to degrade for small values of *loss_p* in the half-lossy scenario (e.g. using the "tail dropping" router this is only 20% when *loss_p* is equal to 0.1). This effect can be seen as an unfairness behavior of TCP when different transmission error rates occur among the sources, since the sources without transmission errors have the tendency to lock-out the sources with transmission errors.

RED and ECN advantage over tail dropping routers: Fig. 15 shows that the advantage of using RED and ECN arises in the half-lossy scenario. In this case, the unfairness effect described in the previous paragraph is reduced. For example, Fig. 15.A shows that when *loss_p* is equal to 0.1 the gain of sources having

transmission errors in the half-lossy scenario is around 45% when RED and ECN is used, while it is only 20% when using tail-dropping. These graphs show also that this fairness benefit of using RED and ECN depends on the delay and the granularity. In fact it can be observed that the higher the granularity, the lower the benefit of using RED and ECN in terms of fairness. Furthermore, the lower is the RTT, the higher is the influence of the granularity. Note that there are other differences between the router algorithms analyzed that cannot be derived from Fig. 15. For example, using tail dropping the queue length has stronger oscillations than using RED. Therefore, the end-to-end transmission delays and their variance are reduced using RED and ECN-RED.

Granularity: the influence of this parameter is twofold since it determines the accuracy in the RTT measurement and the coarse of time-outs. Comparing Fig. 15.A and Fig. 15.B it can be seen that a higher value of the granularity not only reduces the fairness benefit of RED and ECN-RED as explained in the previous paragraph, but also increases the influence of *loss_p* on the gain.

Delay: comparing Fig. 15.A and Fig. 15.C it can be observed that the higher is the RTT, the higher is the influence of *loss_p* on the gain. However, Fig. 15.E shows that this effect is reduced when increasing the number of sources.

Number of sources: comparing the Fig. 15.A- Fig. 15.B respectively with Fig. 15.E-Fig. 15.F it can be observed that the higher is the number of sources, the lower is the reduction of the gain when *loss_p* increases. This is logical since the higher is the number of sources the lower is the average rate obtained for each of them, consequently, the higher is the ratio of discarded (or marked) to transmitted segments by the router in order to adjust the transmission rate governed by TCP. Therefore, the higher loss rate due to transmission errors is needed to interfere with segments discarded (or marked) by the router.

Reno, New-Reno, SACK and FACK: Fig. 16 depicts the gain obtained with each of these TCP implementations in the lossy and half-lossy scenarios using the same parameters than those of Fig. 15.B. The traces obtained using the other sets of parameters of the graphs of Fig. 15 are not shown because similar conclusions that those given in the following where obtained. The first conclusion that can be derived from Fig. 16 is that in the half-lossy scenario, nearly the same result is obtained regardless of the TCP implementation. Therefore, these TCP implementations are not effective to solve the unfairness problem described in the previous paragraphs. Furthermore, from Fig. 16 it can be observed that the influence of the granularity is even higher than the use of a refined TCP implementation. This can be explained since the most important benefit of better TCP implementations is reducing the number of time-outs, but the impact of time-outs is only predominant when the value of the granularity is much higher than the RTT. Finally, from Fig. 16 the following conclusions can be derived: (i) Reno is clearly outperformed by the other TCP implementations, (ii) New-Reno achieves the same performance as the basic SACK implementation (that one we refer to as SACK), (iii) using the more complex TCP implementation (FACK) does not represent a significant improvement over the much simpler New-Reno implementation.

4.3. TCP in a Wireless Network with Micromobility Support

As explained in section 3, several IP micro-mobility protocols have been proposed to enhance the performance of Mobile IP in an environment with frequent handoffs. In this section we make a detailed study of how some of these protocols, namely Cellular IP and Hawaii (see section 2.2.3), affect the behavior of TCP and their interaction with the MAC layer. We investigate the impact of handoffs on TCP by means of simulation traces that show the evolution of segments and acknowledgments during handoffs. A more detailed study can be found in [39].

4.3.1. Simulation Framework

All simulations were conducted using the *network simulator*, ns (see [40, 41]). Fig. 17 shows the topology used in the simulations. This topology consists of two BSs (BS1 and BS2) and a cross-over router. The cross-over router is the Gateway for Cellular IP and the Domain Root Router for HAWAII. The topology was chosen as simple as possible, in order to avoid undesirable complexity and keep the results comprehensible.

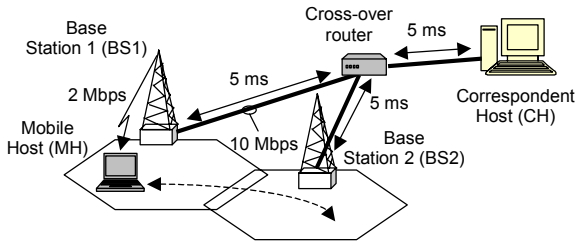


Fig. 17: Simulation testbed.

The Mobile Host moves between BS1 and BS2 such that handoffs are periodically produced. We will refer to the time between two consecutive handoffs as the *handoff period*. There is a cell overlapping of 1 second. BSs send router advertisements every 1 second. The MH uses these router advertisements as beacon signals to recognize the migration from one cell to another, and thus, initiate the handoffs.
In all the simulations a TCP download is simulated. The TCP-Reno implementation is used with a packet size of 1460 bytes and a maximum window size of 20 segments.
Several simulations have been carried out combining the following scenarios:
Radio links: Two kinds of radio links have been used (i) a shared media with an implementation of IEEE 802.11 that works like the 914 MHz Lucent WaveLAN DSSS radio interface, and (ii) an "ideal" wireless interface that consists of a fictitious non-shared media, that is, as if each sender were using a different channel at full link rate (with no collisions). We shall refer to these radio links as the *802.11 MAC* and *Ideal MAC* respectively.
The motivation of using the *Ideal MAC* was to eliminate the effect of a shared media access protocol as the 802.11. This allows to better observe the impact of the micro-mobility protocols. In both cases the bit rate of the radio link was set to 2 Mbps.

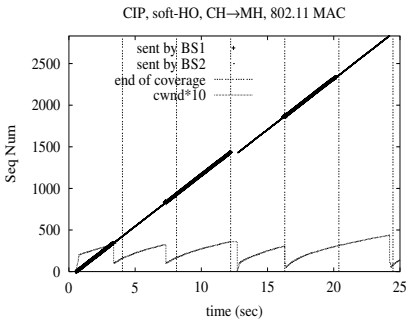


Fig. 18.A TCP segments and congestion window.

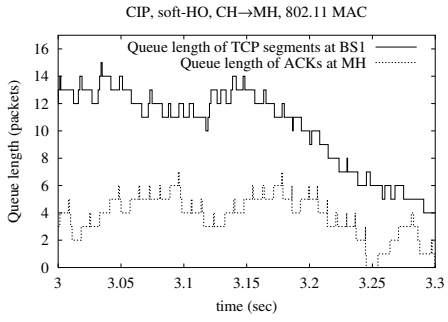


Fig. 18.B Queue length.

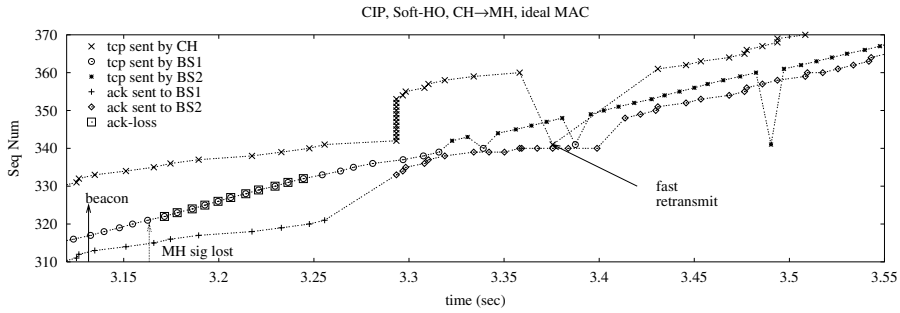


Fig. 18.C Zoom of a handoff.

Fig. 18: CIP with soft handoff traces using a 802.11-MAC

Micro-mobility protocols: we have tested the following micro-mobility protocols using the ns-2 simulator implementation of [41]: (i) Cellular IP with Hard Handoff and Soft Handoff and (ii) HAWAII with MSF and UNF Path Setup Schemes. In the HAWAII-MSF a forwarding buffer with capacity for 20 packets and a time out of 400 ms was used.

4.3.2. Numerical Results

All results discussed in this section have been obtained using the simulation topology described in the previous section. The section is organized as follows: First traces obtained using the 802.11-MAC are shown in order to discuss the impact of the radio link access mechanisms. Then, the dynamics of TCP using each of the micro-mobility protocols (CIP with hard and soft handoff, and HAWAII with MSF and UNF Path Setup Schemes) are presented using the Ideal-MAC.

Impact of the radio link access method

Fig. 5 shows traces obtained in a down-link transmission (from CH to MH) using Cellular IP with soft handoff and a 802.11-MAC radio access link.

Fig. 18.A shows the sequence number of each transmitted segment, the instants when the MH loses the coverage (end of coverage) with the old BS, and the evolution of the congestion window used by TCP (cwnd multiplied by 10 to see it better) at the CH. Fig. 18.B shows the queue length of TCP segments built up at the BS and the queue length of acks at the MH. Fig. 18.C is a zoom of Fig. 18.A. This figure shows: (i) Instants at which the TCP segments are transmitted by the CH (indicated as "tcp sent by CH" in the figure). (ii) Instants at which TCP segments arrive at the MH (indicated as "tcp sent by BS1/BS2"). These instants are marked differently according to the route taken to reach the MH (through BS1 or BS2). Note that these are also the instants when the acks are generated. Transmissions instants of lost acks are marked with a square. (iii) Instants at which acks arrive at the CH (indicated as "ack sent to BS1/BS2"). Again, these instants are marked differently according to the BS used to send them (BS1 or BS2). (iv) The transmission instant of the beacon from the new BS causing the handoff and the transmission instant of the route update message sent by the MH (indicated as *MH sig lost*).

The queue of acks built up at the MH shown in Fig. 18.B seems to be counterintuitive since the 10 Mbps links connecting the CH with the BS are much faster than the 2 Mbps radio link. Therefore, we would expect a queue of TCP segments at the BS, but not the queue of acks at the MH. The reason of this effect is that the acks sent by the MH tend to find the wireless medium occupied by the TCP segments, and their back-off makes that the number of acks per time unit that the MH is able to send into the shared media is lower than the number or TCP segments per time unit sent by the BS. The queue of acks is responsible of the long delay that occurs between the transmission of the ack by the MH and the reception at the CN, as shown in trace (c).

The beacon shown in Fig. 18.C is the first one received from the new BS (BS2). This beacon causes the MH to initiate the handoff. Therefore, the MH switches the radio connection from the old BS (BS1) to the new BS and sends the route update message through the new BS to the gateway router. As indicated in the figure, this route update message and the following 11 acks sent to the new BS are lost. These loses are caused by an address resolution failure motivated by the ack queue built up at the radio link driver of the MH. The IP module at the MH issues an ARP request to the new BS when the route update message is to be sent. The ARP packet is stored at the driver queue and the route update message is kept while waiting for the address resolution. Since the following acks sent by the MH are also addressed to the new BS, and the ARP module keeps only one packet waiting for the resolution of an address, each ack push out the previous packet waiting for the resolution of the new BS address. Only when the ARP query leaves the queue and the address is solved, the following acks are sent to the new BS. The first ack reaching the cross over router changes the routing cache to point to the new BS. During this time the TCP packets are still able to reach the MH through the old BS. These packets are not lost because the MH is able to simultaneously listen to both BSs.

Fig. 18.A shows that although no TCP segments are lost during this first handoff, the TCP sender reduces the congestion window. This is because some of the packets arriving from the old BS reach the MH later than packets reaching the MH through the new BS (as shown in trace (c)). These packets arrive out of order causing the MH to send duplicated acks that trigger the fast retransmit mechanism of TCP. In this case, the fast retransmit of TCP unnecessarily retransmits a packet and reduces the congestion window.

To avoid the described problems caused by the 802.11 MAC influencing the performance of the Micro-Mobility protocols, in the following evaluations we shall use an Ideal MAC.

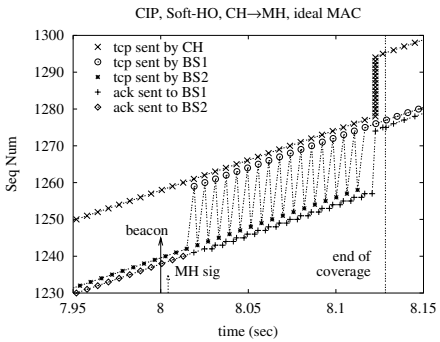


Fig. 19: CIP with soft handoff.

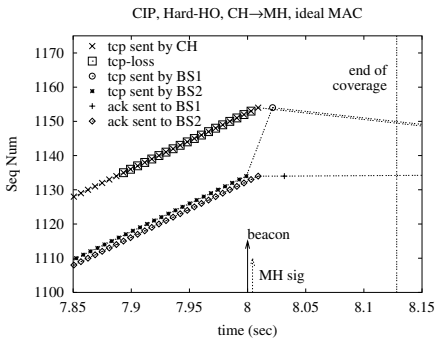


Fig. 20: CIP with hard handoff.

CIP Soft Handoff

Fig. 19 shows the trace obtained with CIP using Soft Handoff. When the MH receives a beacon signal from the new BS, a handoff is initiated while the connection with the old BS is maintained. The MH listens to both BSs during the overlapping of their cell coverage.

The trace shows the transmission instants of the data segments at the CH, and the transmission instants of the data segments at the old BS (BS2) and the new BS (BS1). When the new BS start transmitting TCP segments, the old BS remains transmitting the segments that are enqueued. Since the queue at the new BS is empty, the delay of the segments that go along the path to the new BS is smaller than the ones that go along the path to the old BS. As a result of this, every time a segment transmitted by the new BS arrives at the MH, the MH sends a duplicated ack since the segment is out of order. Only when the last expected packet is transmitted by the old BS, all segments arrived out of order are acknowledged at once and the TCP source sends a whole window of new segments.

CIP Hard Handoff

Fig. 20 shows a trace capturing a handoff obtained with CIP using Hard Handoff. The points depicted in this figure are analogous to those of Fig. 19 but now some TCP segments are lost as shown.

The hard handoff procedure has an important impact on the TCP dynamics. At each handoff, packets get lost when the MH switches the connection from the old BS (BS2) to the new BS (BS1). Packets waiting at the old BS when the connection is switched cannot reach the MH and are lost. This burst of lost TCP segments causes the TCP sender to wait for the retransmission time out and start with a slow start phase. This produces a goodput degradation.

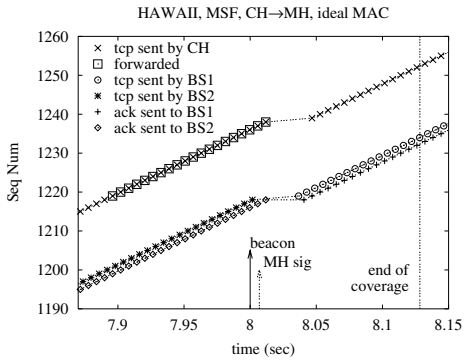


Fig. 21: Hawaii with MSF.

Hawaii with MSF/UNF

Fig. 21 shows the traces obtained using Hawaii with the MSF Path Setup Scheme. In this scenario the MH is able to maintain an ongoing connection with only one BS. Remember that this protocol tries to avoid losing the packets outstanding at the old BS when the connection is switched to the new BS. This is accomplished by forwarding these outstanding packets to the new BS. The trace shows how these forwarded packets effectively avoid TCP packets to be lost, thus solving the goodput degradation that occurs in CIP with hard handoffs.

Using Hawaii with the UNF Path Setup scheme the MH is able to listen to both the new and old BS. This scheme and CIP with soft handoffs have only small differences in the handoff procedures that are not relevant for their performance evaluation, showing similar behavior.

5. Open Research Issues

In this chapter a possible optimization of the Mobility support in IPv6 mechanism has been explored. Nevertheless, additional modifications to the general IPv6 behavior are to be considered in order to improve mobility efficiency. These modifications include different aspects of the specification such as the variation of frequency of Router Advertisement messages and mechanisms for improving micro-mobility support.

Hierarchical Mobile IP combined with a low latency handoff mechanism, such as Pre-Registration of Post-Registration seems to be promising solution to realize seamless handoffs in Mobile IP networks. However, further study is required to investigate how the required layer 2 triggers may be realized for different layer 2 access technologies (such as IEEE 802.11, Hiperlan, etc.). The timing of these triggers may have a major impact on the performance of the handoff and will determine the kind of protocol that is most appropriate, pre-, post-registration of a combination.

TCP has been tuned during many years for networks comprising wired links and stationary hosts. The evolution of TCP will have to take into account the introduction

of wireless technologies, where the assumption of *congestion* as the primary cause of packet losses may not hold anymore. More research and practical experimentation with new wireless technologies need to be performed in order to standardize new TCP implementations that allow differentiating among congestion and other types of packet losses. In wireless networks it is also desirable that ongoing TCP connections do not degrade due to handoffs while the mobile moves within the access network. More research and experimentation is needed to better understand the main causes of packet loss during the handoffs, and the effectiveness to achieve seamless handoffs using the IP micro-mobility protocols that have been proposed in recent years.

References

1. Johnson, D. and C. Perkins, "Mobility Support in IPv6", Internet draft, Work in progress, July 2001.
2. Thomson, S. and T. Narten, "IPv6 Stateless Address Autoconfiguration", RFC 2462, December 1998.
3. Narten, T., Nordmark, E., Simpson, W., “Neighbor Discovery for IP Version 6 (IPv6)”, RFC 2461, December 1998
4. Gruber, J. and Strawczynski, L., “Subjective Effects of Variable Delay in Speech Clipping in Dynamically Managed Voice Systems,” IEEE Trans-actions on Communications, Vol. COM-33, No. 8, Aug. 1985.]
5. Hinden, R., O´Dell, M. and S. Deering, "An IPv6 Aggregatable Global Unicast Address Format", RFC 2374, July 1998.
6. Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 1998, 1998.
7. Schneier, B., "Applied cryptography", Wiley ISBN 0-471-12845-7,1996.
8. Eastlake, D., Crocker, S., Schiller, J., “Randomness Recommendations for security”, RFC 1750, December 1994
9. Narten, T., Draves, R., “Privacy Extensions for Stateless Address Autoconfiguration in IPv6”, RFC 3041, January 2001
10. Montenegro, G., Castelluccia, C., “SUCV Identifiers and addresses”, Internet draft, Work in progress, November 2001
11. Dommety, G., "Fast Handovers for Mobile IPv6", Internet draft, Work in progress, 2001.
12. Wasserman, M., “Recommendations for IPv6 in 3GPP Standards”, Internet Draft, Work in progress, April 2002
13. C. Perkins, ed., “IP Mobility Support”, IETF RFC 2002, October 1996.
14. E. Gustafsson, A. Jonsson, C. Perkins. “Mobile IP Regional Registration”, draft- ietf-mobileip-reg-tunnel-02.txt, March 2000.
15. Hesham Soliman et al., “Hierarchical Mobile IPv6 mobility management (HMIPv6)”, draft-ietf-mobileip-hmipv6-07.txt
16. A. Valko, “Cellular IP – a new approach of Internet host mobility”, ACM Computer Communication Reviews, January 1999
17. Ramjee, R., La Porta, T., Thuel, S., Varadhan, K., and Wang, S., HAWAII: a domain based approach for supporting mobility in wide–area wireless networks, Proceedings of International Conference on Network Protocols, ICNP’99.
18. A. Campbell, J. Gomez, C. Y Wan, S. Kim, Z. Turanyi, A. Valko, “Cellular IP”, IETF draft (draft-ietf-mobileip-cellularip-00.txt), January 2000.
19. A. Campbell, J. Gomez, S. Kim, A. Valko, C.-Y. Wan and Z. Turanyi, “Design, implementation and evaluation of Cellular IP”, IEEE Personal Communications, August 2000, pp.42-49

20. C. Perkins and K-Y. Wang. “Optimized smooth handoffs in Mobile IP”, Proceedings of IEEE Symposium on Computers and Communications, Egypt, July ’99.
21. K. El Malki and others, “Low Latency Handoffs in Mobile IPv4”, IETF draft-ietf-monileip-lowlatency-handoffs-v4-04.txt, 2002.
22. C. Blondia, O. Casals, P. De Cleyn and G. Willems, Performance evaluation of IP micro-mobility solutions, Proceedings Seventh IFIP/IEEE Workshop on Protocols for High-Speed Networks (PfHSN'2002), pp. 211 –226
23. C. Blondia, O. Casals, Ll. Cerdà and G. Willems, Performance analysis of a forwarding scheme for handoff in HAWAII, *Proceedings Networking 2002*, Pisa, Italy, Lecture Notes in Computer Science (LNCS) number 2345. Eds. Enrico Gregori, Marco Conti, Andrew T. Campbell, Guy Omidyar, Moshe Zukerman.
24. C. Blondia, O. Casals, N. Van den Wijngaert, G. Willems,Performance analysis of smooth handoff in Mobile IP, Proceedings MSWiM2002 Performance
25. C. Blondia, O. Casals, Ll. Cerdà, N. Van den Wijngaert, G. Willems, P. De Cleyn, Comparison of Low Latency Mobile IP schemes, to appear in proceedings WiOpt (Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, March 2003
26. O. Casals, Ll. Cerdà, G. Willems, C. Blondia, N. Van den Wijngaert, Performance Evaluation of the Post-Registration Method, a Low Latency Handoff in MIPv4, to appear in Proceedings ICC 2003
27. P. Reinbold and O. Bonaventure, A Comparison of IP mobility protocols, Technical Report infonet-TR-13, 2001, www.infonet.fundp.ac.be
28. B. Braden et al. "Recommendations on Queue Management and Congestion Avoidance in the Internet". RFC 2309, April 1998.
29. K. Fall and S. Floyd. "Simulation-based Comparisons of Tahoe, Reno and SACK TCP". ACM Computer Communication Review, July 1996. ftp://ftp.ee.lbl.gov/papers/sacks.ps.Z
30. S. Floyd and T. Henderson. "The NewReno Modification to TCP's Fast Recovery Algorithm". RFC 2582, April 1999.
31. S. Floyd and V. Jacobson. "Random Early Detection Gateways for Congestion Avoidance". IEEE Transactions on Networking, August 1993.
32. V. Jacobson. "Congestion Avoidance and Control". ACM Computer Communication Review, 18(4), 314-329, August 1988. ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z
33. M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. "TCP Selective Acknowledgment Options". RFC 2018, October 1996.
34. M. Mathis and J. Mahdavi, S. Floyd. "Forward Acknowledgement: Refining TCP Congestion Control". ACM Computer Communication Review, 26(4), October 1996.
35. G. Montenegro, S. Dawkins, M. Kojo, V. Magret, N. Vaidya, “Long Thin Networks”, IETF RFC 2757, Jan. 2000
36. W.R. Stevens. "TCP/IP Illustrated, Volume 1: The Protocols". Addison-Wesley, 1994.
37. G.R. Wright and W.R. Stevens. *TCP/IP Illustrated, Volume 2, the implementation.* Ed: Addison-Wesley, 1995.
38. Ll. Cerdà, O. Casals. “Study of the TCP Unfairness in a Wireless Environment”. Proceedings of IEEE ICT 2001, Bucharest, Rumania, June 2001.
39. F. Vena, Ll. Cerdà, O. Casals. “Study of the TCP Dynamics over Wireless Networks with Micromobility Support Using the ns Simulator”. European Wireless 2002, Florence, Italy, February 2002.
40. The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns
41. ns with micromobility support, http://comet.ctr.columbia.edu/micromobility

Algorithms for Scalable Content Distribution

Ernst W. Biersack (Ed.)[1], Anwar Al Hamra[1], Guillaume Urvoy-Keller[1], David Choi[1], Dimitrios N. Serpanos[2], and Apostolos Traganitis[3]

[1] Institut Eurecom, Departement of Corporate Communications
B.P. 193, 06904 Sophia Antipolis, France
{erbi, alhamra, keller, choi}@eurecom.fr
[2] University of Patras
Department of Electrical and Computer Engineering
serpanos@ee.upatras.gr
[3] University of Crete, Department of Computer Science
tragani@csd.uoc.gr

Abstract. In this chapter, we address how to achieve scalable content distributions. We present two contributions, each of which uses a different approach to distribute the content.
In the first part of this chapter, we consider a terrestrial overlay network and build on top of it a VoD service for fixed clients. The goal is to minimize the operational cost of the service. Our contibutions are as follows. First, we introduce a new video distribution architecture that combines open-loop and closed-loop schemes. This combination makes the overall system highly scalable, very cost-effective, and ensures a zero start-up delay. Our second contribution is a mathematical model for the cost of delivering a video as a function of the popularity of that video. Our analytical model, along with some extensions, allows us to explore several scenarios: (i) long videos of 90 min (movies), (ii) short videos of a few min (clips), (iii) the dimensioning of a video on demand service from scratch, and (iv) the case of the optimization of an already installed video on demand service (i.e. the limited resources scenario).
In the second part of this chapter, we consider a satellite distribution of contents to mobile users, or in general to users thar are occasionally connected. We consider a push-based model, where the server periodically downloads objects. We assume that clients register to the service off-line. Our goal is to minimize the mean aggregate reception delay over all objects where each object is weighted by its popularity. Our contibutions in this part are as follows. First, we provide a simple proof for the need of periodicity (equal distance in transmission) of popular objects in a cycle. Second, in contrast to existing results, we consider the scheduling problem for caching clients. To increase the performance of the system, we also evaluate a pre-emptive scheduling algorithm that allows interruption (pre-emption) of an object's transmission in order to transmit on schedule another more popular one.

[1] Institut Eurécom's research is partially supported by its industrial members: Bouygues Télécom, Fondation d'entreprise Groupe Cegetel, Fondation Hasler, France Télécom , Hitachi, ST Microelectronics, Swisscom, Texas Instruments, Thales

M. Smirnov et al. (Eds.): COST 263 Final Report, LNCS 2856, pp. 207–262, 2003.

1 Introduction

The Internet has evolved, in the last decade, from a research oriented network to a large scale commercial network. Still, most of the trading concerns non-real time documents or services and only two parties are involved in general (dealer/client). The deployment of content distribution networks faces two main obstacles:

- There currently exists no scalable and widely deployed means to serve simultaneously a great number of clients;
- The wide heterogeneity of clients, not only in terms of access bandwidth, but also in terms of reachability since clients might be fixed or mobile.

To achieve a scalable distribution (whose cost grows less than linearly with the number of clients), two approaches have been commonly used: (i) a dedicated overlay network to provide a multicast service at the application layer or (ii) a satellite distribution whose cost is essentially independent from the number of simultaneous clients. In this chapter, we present two contributions that use either one of these scalable distribution techniques with different hypotheses concerning the clients:

In the first part of this chapter, we use a terrestrial overlay network and build on top of it a video on demand service for fixed clients. Our objective is to minimize the operational cost of the service. To minimize the distribution (bandwidth) cost, we use an open-loop central server since its cost is roughly independent of the number of simultaneously connected clients. The only limitation of open-loop algorithms is that they induce a non-zero start-up delay, which accounts for the duration between the moment where the client starts receiving the video stream and the moment it is able to visualize the video on the screen. To achieve a zero start-up delay, we thus introduce servers that stream the beginning of the video (which is referred to as prefix) immediately upon connection of the client to the video on demand service. To build a cost-optimal video on demand service, we thus end up solving an optimization problem whose variables are: the popularity of the requested video, the number of video servers that we use to service the beginning of the video and the fraction of the video that the "prefix" represents.

Our contribution is the design of a mathematical model that allows to find the architecture (i.e. the optimal values of these variables) that minimizes the total operational cost of our video on demand service. In contrast to previous studies, our model considers realistic underlying physical network infrastructures to accurately model all bandwidth costs and also includes both the I/O and storage requirements. Our analytical model, along with some extensions, allows us to explore several scenarios: (i) long videos of 90 min (movies), (ii) short videos of a few min (clips), (iii) the dimensioning of a video on demand service from scratch, and (iv) the case of the optimization of an already installed video on demand service (i.e. the limited resources scenario).

In the second part of this chapter, we consider a completely different setting since we assume a satellite distribution of content to mobile users that are occasionally connected. We consider a push model where a client has registered off-line to a service that can be modeled as the periodic download of objects representing, for instance, pages, pictures, or short videos. The objective is to minimize the mean aggregate reception delay of all objects where an object is weighted by its popularity. This metric is important since it ensures minimization of the energy consumption at the client side. Existing studies on broadcast systems generally assume that a client is not able to temporarily cache an object. In contrast to these studies we consider the case where clients are equipped with a cache, which is a realistic assumption with respect to the recent evolution of technology.

Our contributions in the second part of this chapter are the following. We first derive a simpler proof than the existing one justifying the periodic transmission of objects during a cycle is optimal for the case of cache-less clients. We then extend this proof to the case of caching clients. We also devise the corresponding optimal algorithm that computes the optimal schedule of each object. In practice, however, achieving a perfect periodicity of the broadcasting of objects within a cycle is an NP-hard problem because the optimal time to transmit an object may lie before the end of transmission of the previous one.

To improve the performance of the system, we investigate several empirical pre-emption strategies, some of them resulting in a real performance improvement while others, like the ones based on interleaving transmissions of multiple objects are proved to be inefficient.

2 Cost-Optimal Dimensioning of a Large Scale Video on Demand System

2.1 Introduction

Background Video streams such as MPEG-2 encoded video require several Mbps and providing a VoD service to a large number of clients poses high resource demands to the server and the network. The bandwidth-intensive nature of video requires efficient distribution techniques that typically serve multiple clients who request the same video at approximately the same time via a single video stream that is multicast. VoD systems can be classified in *open–loop systems* [6, 43, 2, 25, 32, 33, 11] and *closed–loop systems* [3, 39, 26, 19, 37, 17].

- Open-loop VoD systems partition each video into smaller pieces called *segments* and transmit each segment at its assigned transmission rate. The first segment is transmitted more frequently than later segments because it is needed first in the playback. All segments are transmitted periodically and indefinitely. In open–loop systems there is no feedback from the client to the server and transmission is completely one–way. Open-loop systems are suitable for popular videos where multicast is efficient. However, open-loop

systems introduce a start-up delay[4] and waste bandwidth in the case of non-popular videos.
- Closed-loop systems, on the other hand, require the client to contact the server. Closed–loop systems generally open a new unicast/multicast stream each time a client or a group of clients issue a request for a video. Whenever a new client arrives, the client joins an ongoing multicast stream that has been initiated for earlier clients, if there is any, and retrieves the missed part due to its later arrival via unicast, which ensures an immediate playout of the video. More often, Closed-loop schemes are suitable for videos with moderate popularity where the load on the server is reasonably low.

In the first part of this chapter, we present a new video distribution architecture that combines both, open-loop and closed-loop systems. This combination makes our video distribution architecture suitable for both, popular and non-popular videos. Moreover, having used a closed-loop system to deliver the first part of the video, our video distribution architecture ensures a zero start-up delay.

Contributions and Related Work Providing an efficient and scalable video distribution to a large client population has been investigated extensively over the years. The basic idea to achieve scalability is to serve multiple clients via a single stream, using multicast. Open-loop schemes take a full advantage of multicast. *Staggered broadcasting* [6] is the straightforward open-loop scheme where the server allocates for each video C channels each of bandwidth equal to the play back rate b of the video. On each channel, the whole video is broadcast at rate b. The starting points of the transmission on the different channels are shifted to guarantee a start-up delay of no more than L/C, where L is the length of the video. Clients listen to only one channel at the same time and no storage capacity is needed at the client side. The start-up delay can be improved only by increasing the number of allocated channels C.

More efficient and complicated schemes have been proposed later on. *Pyramid Broadcasting* (PB) [43] divides the video into C segments of increasing size, where C is also the total number of logical channels. The size of each segment is made α times larger than the size of the previous segment ($L_i = \alpha L_{i-1},\ i > 1$). The value of α is set to $\frac{B}{K \cdot C}$, where K is the total number of videos in the system and B is the total bandwidth over all the C channels. Note that B is divided equally amongst the C channels. Segments i for all videos are multiplexed into the same channel i and broadcast consecutively and periodically on that channel i at rate B/C. At any moment, the client downloads from at most two channels. This scheme reduces the bandwidth utilization and the start-up delay as compared to the *staggered broadcasting* [6] while guaranteeing a continuous playback of the video. However, its main disadvantage is that segments are transmitted on the different channels at a high rate (i.e. B/C) which requires a high I/O capacity at the client side.

[4] We ignore here the *transmission* delay due to sending a request to a server or joining a multicast group.

Permutation-based Pyramid Broadcasting (PPB) [2] addresses the problem of high I/O requirement at the expense of a larger start-up delay and a more complex synchronization. PPB uses the same geometric series proposed by PB to define the length of each segment ($L_i = \alpha L_{i-1},\ i > 1$). Unlike PB, PPB proposes to divide each channel into $K \cdot p$ sub-channels, where p sub-channels are dedicated for the same video segment. Then, each segment is broadcast on p sub-channels in such a way that the access time of segment i is L_i/p. Thus, segments are transmitted at rate $B/(C \cdot K \cdot p)$ instead B/C in the case of the PB scheme. On the other hand, at any moment, clients download from at most two separate sub-channels.

Another efficient scheme is the *skyscraper Broadcasting* (SB) [25]. SB uses a recursive function instead of a geometric series to generate the segment lengths. Each segment is then broadcast on a separate channel at the playback rate b. Clients listen to at most two channels at the same time. This scheme accounts for buffer space limitations of the clients by constraining the size of the last segment.

Fast Broadcasting (FB) [32] divides each video into segments according to a geometric series ($L_i = 2^i$). Each segment is broadcast on a separate channel at the playback rate b. Clients listen to all channels at the same time. FB provides the lowest bandwidth requirement at the server as compared to the above schemes.

Harmonic Broadcasting (HB) [33] presents another variant from the same general idea. In this scheme, the video is divided into N segments of equal size. Segment i is transmitted at the rate $1/i$ and clients download from all channels at the same time.

All the schemes cited above are constrained to highly regular designs which limits their flexibility. The *Tailored Transmission Scheme* [11] is a more flexible organization that can be adapted to meet different constraints in the system such as limited I/O capacities of the server/clients, limited storage capacity at the client side. This scheme will be detailed more in subsubsection 2. For more details on open-loop schemes, see [24].

While open-loop schemes broadcast the video regardless the request pattern of the clients, closed-loop schemes serve the video in response to client requests. Closed-loop schemes can be classified into batching, patching, and hierarchical merging schemes. The basic idea of batching [3, 39] is that the server groups clients that arrive within a given interval of time, in order to serve them via a single multicast stream. However batching introduces a start-up delay.

Patching techniques [26], on the other hand, ensure a zero start-up delay. The first client that requests a video receives a complete stream for the whole video from the server. When a new client arrives after the first one, we distinguish two cases:

– The complete stream that has been initiated for the first client is still active. In this case, the client needs to connect to that stream which changes from unicast to multicast. In addition, the client receives immediately from the

server a unicast patch for the part it missed in the complete stream due to its late arrival.
- There is no active complete stream. In this case, the server initiates a new one and the process repeats for clients that arrive later.

Note that clients that are listening to the same complete stream form a session. One efficient extension of patching has been proposed in [19] with the introduction of a threshold policy to reduce the cost of the unicast patches. Whenever a new client arrives and a complete stream is active, the threshold value serves to decide whether to initiate a new complete stream for that client, or whether that client must join the last ongoing complete stream. This scheme will be explained more in subsubsection 2.

White et al. [45] propose OBP, a hybrid scheme that combines patching and batching techniques. As for the classical patching [26], in OBP, the server initiates a new complete stream for the first client. A later client either receives a new complete stream in case there is no active complete stream, or the client joins an already existing one. In this later case, in contrast to patching, the client does not receive immediately the missed part in the complete stream; instead, the server batches many clients that arrive within a given interval of time and serve them via multicast. Thereby, as compared to patching, OBP reduces the delivery cost of the missed parts at the expense of a larger commencement viewing delay of the video. Similar to the controlled multicast, the authors introduce a threshold to decide when a new session should start. They also extend the OBP to deal with the case of heterogeneous clients where each client chooses the start-up latency according to the price it is willing to pay for the service.

Hierarchical merging [17] is a more efficient closed-loop scheme. As its name indicates, this scheme merges clients in a hierarchical manner. When a new client arrives, the server initiates a unicast stream to that client. At the same time, the client listens to the closest stream (target) that is still active. When the client receives via unicast all what it missed in the target stream, the unicast stream is terminated and the client merges into the target stream, and the process repeats. It has been shown that the server bandwidth for the hierarchical merging increases logarithmically with the number of clients.

In this work, we propose a scalable and efficient video distribution architecture that combines open-loop and closed-loop mechanisms to assure a zero start-up delay. Each video is partitioned into a prefix and a suffix. The suffix is stored at a central server, while the prefix is stored at one or more prefix servers. A client who wants to view a video joins an already on-going open-loop multicast distribution of the suffix while immediately requesting the prefix of the video as a patch [26] that is sent either via unicast or multicast [19]. We develop an analytical model for that video distribution architecture that allows to compute for a video with a given popularity the cost-optimal partitioning into prefix and suffix and the placement of the prefix servers in the distribution tree.

In contrast to previous studies (see for example [13, 21, 44]), we

- Model the network as a tree with outdegree m and l levels. In comparison, Guo et al. [21] consider only a two-level distribution architecture.

- Account in the model of the network transmission cost for the number of clients that are simultaneously served by the multicast distribution (either from the prefix servers or the suffix server).
- Allow for the prefix servers to be placed at any level in the distribution tree and not only at the last hop between client and network [44].
- Include in our cost model not only the network transmission cost but also the *server* cost, which depends on both, the storage occupied and the number of input/output streams needed. While the network transmission cost is a major cost factor, the server cost must be included in the overall cost model, especially when we try to design a cost-optimal video distribution architecture. Otherwise, independent of the popularity of a video, the obvious/trivial architecture will be the one where a large number of prefix servers are placed near the clients. While previous papers [21] have treated in their model the storage space of the prefix servers as a scarce resource, we feel that the cost model can be made more realistic by explicitly modeling the cost of the prefix servers.

A related cost model has been presented previously in [35]. The authors model the distribution network as a tree with l levels. In this model, *caches* can be placed at any level in the network. However, these caches can only store the entire video. Our model is more flexible in the sense that any portion of the video can be stored at the prefix servers. The system cost is simply the sum over the storage cost, the I/O bandwidth cost of server/caches, and the network bandwidth cost. Moreover, the cost formulas developed are for simple scenarios with only unicast server/clients connections.

A recent paper by Zhao et al. [48] looks at different network topologies, such as fan-out K, daisy-chain, balanced tree and network topologies generated with topology generators. The analysis focuses on schemes that provide an instantaneous delivery of the video. They derive a tight bound on the minimum network bandwidth requirement for each topology. They show that, at best, the minimum network bandwidth requirement scales as $\mathrm{O}(lnN)$, where N is the average number of clients during an interval of time equal to the duration of the video. They also show that it is possible to achieve simultaneously close to the minimum network and server bandwidth usage with practical distribution protocols.

The rest of this sub-chapter is organized as follows: in Subsection 2.2, we present the broadcasting algorithm and content distribution overlay that we use to build the video distribution service. In Subsection 2.3, we present the basic PS-model that allows to obtain the cost formulas for the single video case when there is no a priori constraint on the content distribution network. In Subsection 2.4, we apply the PS-model to the case of long videos, and investigate different scenarios such as heterogeneous or homogeneous bandwidth costs, zero servers costs, etc.. In Subsection 2.5, we prove that the PS-model is still advantageous in the case of short videos, e.g. news clips. In Subsection 2.6, we first study the dimensioning of the system with n videos and no resource constraints. We next devise the algorithm that optimally assigns video to prefix servers in the

case of limited resources (I/O, storage) and fixed placement of these servers. We conclude this sub-chapter in Subsection 2.7.

2.2 The System Environment

Prefix Caching Assisted Periodic Broadcast Prefix caching assisted periodic broadcast[5] [21] assumes that clients are serviced by a main central suffix server and also by local prefix servers, which can be located throughout the network. A video is partitioned into two parts, the prefix and the suffix, which can be of arbitrary proportion. We denote by L (min) the length of the video and D (min) the length of the prefix. Hence, the suffix will have the length $L - D$. The entirety of the prefix is always *viewed before* the suffix. The main idea of the broadcast scheme is that prefix and suffix transmissions should be decoupled in order to transmit each most effectively. The reason why the prefix and suffix are transmitted differently is that the client must receive the prefix *immediately upon request* while the suffix needs not to be received until the prefix has been completely viewed.

Because the prefix must be immediately received, there is less flexibility in the choice of a transmission scheme for the prefix. In addition, transmitting the prefix from the central server to each client may be costly. In order to reduce transmission costs, the prefix can be stored locally at multiple prefix servers, which can more cheaply transmit the prefix to their local audiences. For the suffix, on the other hand, there is more leeway in the method of broadcast since it needs not to be received immediately. The allowable delay D in transmitting the suffix permits to deliver it with an open-loop scheme. Therefore, the suffix is retained at the central server, benefiting from sharing amongst a relatively larger number of clients as well as avoiding the server costs incurred to replicate data across multiple servers.

Once specific transmission schemes for prefix and suffix have been chosen, the remaining design parameters are the length of the prefix (and suffix) and the height of placement of the prefix servers in the network. The prefix length and the location of the prefix servers should be chosen so as to efficiently divide the workload between central suffix server and prefix servers.

In our prefix caching assisted periodic broadcast model, we choose to transmit the prefix via *controlled multicast*, while the suffix is delivered through *tailored periodic broadcast*.

The Distribution Network We assume that the distribution network is organized as an *overlay* network. An overlay network consists of a collection of nodes placed at strategic locations in existing network, i.e. the Internet. Overlay networks provide the necessary flexibility to realize enhanced services such as

[5] The term broadcast is commonly used in the literature. In our model, broadcast really refers to multicast as the data that are sent only over links that reach clients who are interested in receiving the video.

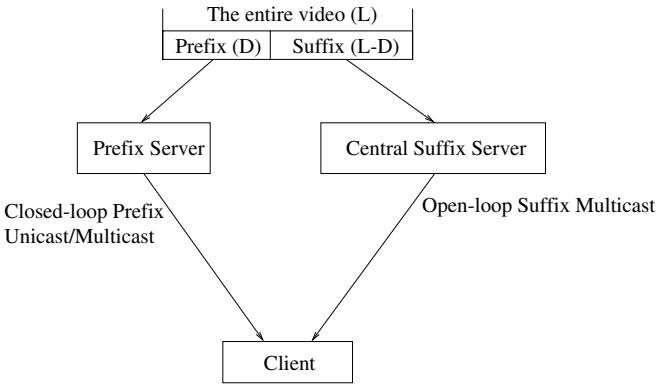


Fig. 1. The VoD distribution architecture

multicast [31] or content distribution [1] and are typically organized in a hierarchical manner.

In our model, we assume that the topology of our distribution network is a m-ary tree with l levels (see figure 2). The suffix server is assumed to be at the root. The prefix servers may be placed at any level of the distribution network other than the highest level (i.e. leaves). If the prefix servers are placed at level j, there will be one at each node of that level, which makes a total of m^j prefix servers. The clients are lumped together at the m^l leaf nodes. The number of clients watching simultaneously a video is not limited to m^l since a leaf node does not represent a single client but multiple clients that are in the same building.

We digress briefly to consider the practical aspects of a network tree model. A tree model captures the hierarchical structure of a large-scale network, where large backbone routers service many smaller service providers which in turn service the end-user clients. For example, a tree might include multiple levels, dividing the network into national, regional and local sub-networks.

The distribution network is assumed to support both unicast and multicast transmissions. Unicast transmission occurs between a server and a single client, whereas multicast transmission occurs when multiple clients (possibly from different leaf nodes) all simultaneously receive the same single transmission from a server. We assume that for the duration of a transmission, a cost must be paid only for every link spanned between the server and its active client(s). The per-link cost may differ depending upon the specific links that are utilized.

For a multicast transmission, the cost may change over the duration of the transmission as users join and leave. Note also that if multiple clients reside at a single leaf node then the cost of multicast transmission is effectively the same as if there were only a single client at that node. For clients at different nodes, multicast still offers savings due to links shared at the lower levels by different nodes.

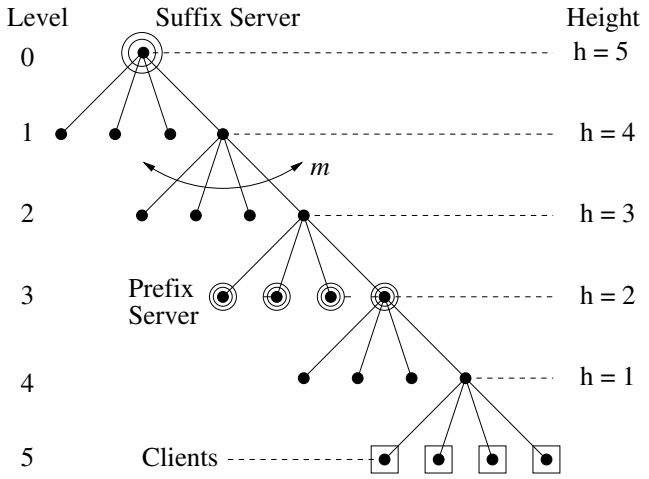


Fig. 2. Video distribution network

Prefix Transmission via Controlled Multicast Patching was first proposed in [26] and then extended with the inclusion of a thresholding policy to produce **Controlled Multicast** [19]. The key idea of patching is to allow clients to share segments of a video stream when they arrive at different times. As the number of clients increases from one to several, the transmission stream is changed from a unicast stream to a multicast one so that late arrivals can still share in the remainder of the stream. In addition, a separate unicast stream must also be transmitted to each client after the first one in order to deliver the data missed due to its later arrival.

For extremely late arrivals, the cost of the additional unicast transmission may outweigh the benefits of sharing in the remaining transmission. Controlled multicast modifies patching to allow for this scenario. Whenever a new transmission is started at time t, arriving clients are patched onto the stream until time $t+T$, where T is a **thresholding** parameter. The first client to arrive after time $t+T$ is given a brand new transmission, and all future arrivals are patched onto the new transmission instead of the old one, until the threshold time passes again and the process is repeated. Figure 3 illustrates the operation of controlled multicast. At time t_1 the first client arrives: The prefix server starts transmitting the prefix via unicast. When the second client joins at time t_2, the remaining part of the prefix is multicast to clients 1 and 2. Client 2 additionally receives the initial part of the prefix that had been transmitted between t_1 and t_2 via a separate unicast transmission. Since client 3 arrives at time t_3, with $t_3 - t_1 > T$, the prefix server starts a new unicast transmission of the entire prefix.

The costs of controlled multicast have been shown to increase sub-linearly with the arrival rate of requests and the length of the prefix [37]; however, the analysis assumes a network of a single link between the server and all its clients. This is the case when the prefix servers are located one level above the clients.

We refer to that case as **leaf prefix server placement**. Placing the prefix servers higher up in the distribution network increases the cost of transmission; however, it consolidates the arrivals to a smaller number of prefix servers and thereby allows for more sharing to occur amongst clients. Furthermore, placing the prefix servers higher up in the network reduces server costs since there are fewer copies of the prefix. One contribution of this paper is an analysis of the tradeoffs in prefix server placement for controlled multicast.

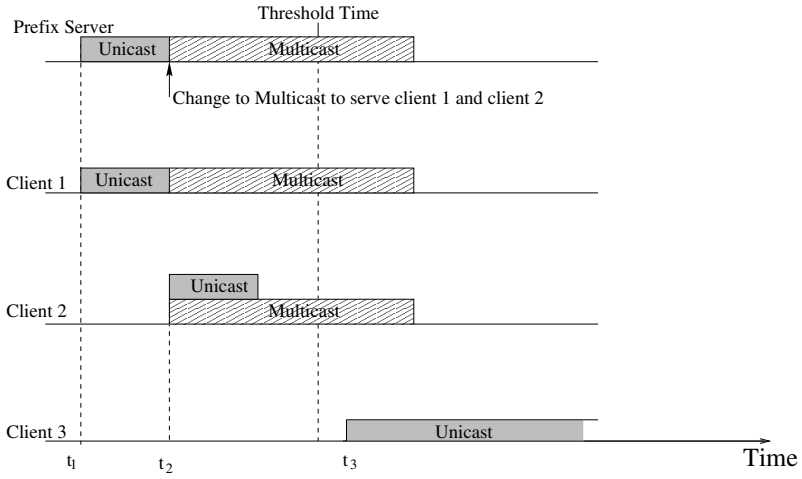


Fig. 3. Prefix transmission with multicast and patching

Tailored Periodic Broadcast of the Suffix We use tailored transmission [11] to transmit the suffix. Thereby, we divide the suffix into segments of fixed lengths. If there are no clients then the suffix server does not transmit. As long as there is at least one client, each segment is periodically multicast at its own transmission rate. Arriving clients receive the multicast of each segment simultaneously. Clients are not expected to arrive at the starting point of each segment; instead, they begin recording at whatever point they arrive, store the data and reconstruct each segment as they receive the data.

The length and rate of each segment is chosen so as to minimize the bandwidth subject to the constraint that each segment must be completely received before its time of playback, and also subject to the storage and reception capabilities of the clients. Figure 4 illustrates the tailored transmission scheme for the case of minimal transmission rates. The client starts receiving all segments at time t_0. The shaded areas for each segment contain exactly the content of that segment as received by the client who started recording at time t_0.

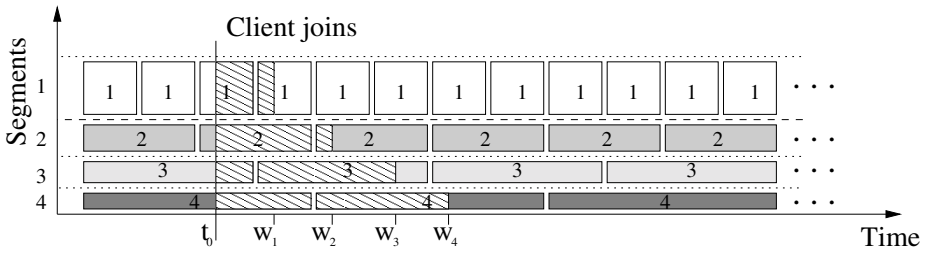


Fig. 4. Tailored Broadcast transmission

The total server transmission bandwidth is:

$$R_t^{min} = \sum_{i=1}^{N_s} r_i^{min}$$

where $N_s \triangleq \lceil \frac{L-D}{D} \rceil$ is the number of segments the suffix consists of and r_i^{min} is the minimal transmission rate of segment i. When we break the suffix of length $L - D$ into segments, each of the segments 1 to $N_s - 1$ has length D and the last segment N_s has a length of $(L - D) - (N_s - 1)D = L - N_sD$. Segment i with length D needs to be entirely received no later than $iD \cdot 60$ seconds after the start of the video consumption. The factor 60 in $iD \cdot 60$ is due to the fact that the lengths of the video, the prefix, and the suffix are expressed in *minutes*. Therefore, the minimal transmission rate for segment i is $r_i^{min} = b\ \frac{D \cdot 60}{iD \cdot 60} = \frac{b}{i}$, where b [Mbit/sec] is the consumption rate of the video and $bD \cdot 60$ is the amount of data [Mbit] in a segment of length D.

In the case where the length of the last segment N_s is less than D (i.e. $\frac{L-D}{D}$ is not an integer), the transmission rate $r_{N_s}^{min}$ is computed as follows depending on whether N_s is larger than 1 or not.

- If $N_s > 1$, the segment N_s should be entirely received $N_sD \cdot 60$ seconds after the start of the video consumption. As we will see later, we assume that all the segments are multiplexed onto a single multicast channel and the receiver stays tuned into that multicast channel until it has received the last segment. Therefore, clients will receive the first segments multiple times. Thus, in this case where the length of segment N_s is less than D, it might be efficient to reduce the tuning time of clients. This can be done by reducing the transmission time of segment N_s to $(N_s - 1)D \cdot 60$ instead $N_sD \cdot 60$ seconds. For a video with a low demand, reducing the service time of the client to $(N_s - 1)D \cdot 60$ increases the transmission rate r_{N_s} of the last segment; however, the increase is offset by avoiding useless transmissions of some of the first segments. For a popular or very popular video, the prefix length is quite short which means that $N_s \sim (N_s - 1)$, and so the increase in r_{N_s} has a negligible impact on the overall server transmission bandwidth R_t^{min}. So, using the fact that $N_s = \lceil \frac{L-D}{D} \rceil = \lfloor \frac{L}{D} \rfloor$, the transmission rate of segment N_s then becomes $r_{N_s}^{min} = b\ \frac{(L-N_sD) \cdot 60}{(N_s-1)D \cdot 60} = b\ \frac{(\frac{L}{D} - \lfloor \frac{L}{D} \rfloor)D}{(N_s-1)D} = b\ \frac{\frac{L}{D} - \lfloor \frac{L}{D} \rfloor}{N_s-1}$.

– If $N_s = 1$, the suffix consists of one segment of length $(L - D) < D$ that must be entirely received $D \cdot 60$ seconds after the start of the video consumption. The transmission rate of segment N_s then becomes $r_{N_s}^{min} = r_1^{min} = b \, \frac{(L-D)\cdot 60}{D \cdot 60} = b \, \frac{L-D}{D}$.

The total server transmission bandwidth is therefore:

$$R_t^{min} = \begin{cases} b \, (\sum_{i=1}^{\lfloor \frac{L-D}{D} \rfloor} \frac{1}{i} + \frac{\frac{L}{D} - \lfloor \frac{L}{D} \rfloor}{N_s - 1}) & \text{if } N_s > 1 \\ b \, \frac{L-D}{D} & \text{if } N_s = 1 \end{cases}$$

Transmitting all the segments at their minimal transmission rate requires that the client starts receiving all segments simultaneously. As a consequence, parts of the video will be received some time before they are consumed and must be stored by the client in the meantime. Figure 5 plots the evolution with time of the amount of data stored for a video of length $L = 90$ min and a prefix length of $D = 2$ min. We see that at the beginning, more and more data will be received ahead of time so that the amount of data keeps increasing until it reaches a peak corresponding to nearly 40 percent of the video. From there on, data will be consumed at a rate higher than the aggregate rate at which new data are received and the storage curve decreases. Storing up to 40 percent of the video data does not pose any problem with todays equipment. In fact, there already exist products such as the digital video recorder by TiVo [40] that can store up to 60 hours of MPEG II encoded video.

The tailored periodic broadcast scheme also allows to support user interactions [10] such as fast forward if the transmission rate of each segment is increased by a small factor (less than twice the minimal transmission rate), provided that the client has enough buffer to store large parts of the video.

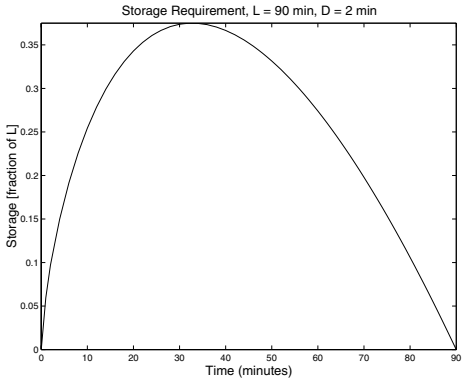


Fig. 5. Storage required at client as function of time, $L = 90, D = 2$

Interaction between Clients and Servers When a new client wants to receive a video, it will contact its closest prefix server in the distribution tree. The prefix server will either start a new transmission cycle of the prefix or extend the ongoing prefix multicast transmission to the new client and transmit a unicast patch to the client. However, the client does not need to contact the suffix server. The suffix server simply needs to know whether there is currently at least one client who needs to receive the suffix, which the suffix server can learn by communicating with the prefix servers. Other large scale video distribution schemes such as hierarchical stream merging [18] require that all client requests be handled by the central server, which makes the central server a potential bottleneck. Our video distribution scheme is not only *scalable* in terms of the network and server resources required to stream a video to a large number of clients but also in terms of processing incoming client requests.

2.3 PS-model, a Cost Model for the Video Transmission

Introduction We divide the costs of a VOD network into network and server costs. The network costs are proportional to the amount of network bandwidth which is transmitted over each link between a server and its clients. The server cost is dependent upon the necessary storage and upon the total number of input/output streams which the server(s) must simultaneously support over the network.

It is interesting to note that the storage and input/output stream capacity of a server cannot be purchased in arbitrary quantities. Instead, they can usually only be purchased in discrete increments. As a result, it is unlikely that a server can be purchased to exactly match both storage and streaming requirements; typically one constraint will be slack as the server will either have extra storage or extra streaming capability.

We will examine the expected delivery cost for a video of length L as a function of the average request rate per *minute* λ (1/min), the prefix length (and allowable suffix delay) D and the topology of the network. The maximum output capacities should be fixed for each server; however, to facilitate the analysis we will assume that any number of streams can be allocated with the costs paid on a per-stream basis.

We divide the multicast tree into levels $1, \ldots, l$, where level 1 consists of the m links from the root and level l consists of the m^l links connected to the leaf nodes. Arrivals to a link at level j are modeled as a Poisson process with parameter λ/m^j. As a consequence, arrivals at the root will form a Poisson process with parameter λ.

We first derive the cost of prefix transmission with a single prefix server at the root. We next generalize the root case to the general case where the prefix servers are placed at some level between the root and the clients. The costs fall into three categories: network, storage capacity and I/O capacity.

We neglect the effects of network latency, even though they can be important from an operational point of view. One might treat latency effects by constraining the maximum number of hops allowed between prefix servers and their clients.

We will derive the cost for the prefix and for the suffix transmission separately and then combine both to obtain the overall system cost. We will refer to this cost model also as the **PS-model** to distinguish it from other cost models.

Costs for Prefix Transmission

Prefix server at the root We first consider a single prefix server at the root of the multicast tree. We will afterwards generalize our results to the case where the prefix server is at an arbitrary height in the tree.

Network bandwidth costs:

A single server at the root combines many small request arrival streams (to the leaves) into a single large arrival stream (to the root); this should lower costs since the cost of prefix transmission via controlled multicast is sub-linear in the arrival rate because it promotes efficiency through shared streams; by combining all the requests arriving at the root, we increase the possibility for sharing. However, this is counterbalanced by the fact that sharing is no longer free; if two clients share the same I/O stream but reside on different leaves, a separate network cost must be paid for each of them. Of course, if clients are already active at every leaf node, then no new network costs must be paid for any future arrivals. However, this scenario is unlikely even for high arrival rates because high arrival rates produce short threshold times in order to reduce the length of the unicast streams.

Let t_i be the time of the ith complete multicast transmission of the prefix without any patching. Arrivals between times t_i and t_{i+1} will share from the multicast transmission at time t_i and will each receive a separate unicast transmission for the data which were missed. We can divide the patching process up into separate renewal cycles $(t_1t_2], (t_2t_3], \ldots$ which are independents and identically distributed in their usage of bandwidth. We analyze the bandwidth usage over a single renewal cycle.

Given the threshold time T, on average there will be $T\lambda$ arrivals which will each need partial transmission of the prefix in unicast. The average length of the unicast transfer will be $T/2$ since the arrivals are uniformly distributed over time. Finally a bandwidth cost must be paid for every link on the path between client (at the leaf) and the root server. As a result, the total amount of data transmitted for the unicast streams over one renewal cycle is

$$C_{netw}^{unicast} = b \cdot 60 \, \frac{lT^2\lambda}{2}.$$

Each arrival will also share from a single multicast network stream. A price must be paid for every link in use. Given a link at level j, let τ_j be the duration of time in which the link is active. For the multicast stream, a link is active from the time of the first arrival (before time T) to that link to the end of the prefix at time D. Arrivals to a link at level j form a Poisson process with parameter λ/m^j.

As each renewal cycle begins with the activation of a stream between the root and a single client, we know that one link at each level will be active at

time zero. Therefore $\tau_j = D$ with probability $1/m^j$. We will now write out an expression for $\mathbb{E}[\tau_j]$:

$$\begin{aligned}\mathbb{E}[\tau_j] &= D\mathbb{P}\{\tau_j = D\} + \mathbb{E}[\tau_j|\tau_j \neq D]\mathbb{P}\{\tau_j \neq D\} \\ &= D\frac{1}{m^j} + \mathbb{E}[\tau_j|\tau_j \neq D]\frac{m^j - 1}{m^j}.\end{aligned}$$

Given a Poisson process with parameter λ/m^j, the time of first arrival will have an exponential distribution with parameter λ/m^j and a cumulative distribution $F(t) = 1 - e^{-\frac{\lambda}{m^j}t}$. We evaluate $\mathbb{E}[\tau_j|\tau_j \neq D]$ making use of the fact that $\int_0^T t\frac{\lambda}{m^j}e^{-\frac{\lambda}{m^j}t}dt = \int_0^T (e^{-\frac{\lambda}{m^j}t} - e^{-\frac{\lambda}{m^j}T})dt$.

$$\begin{aligned}\mathbb{E}[\tau_j|\tau_j \neq D] &= \int_0^T (D-t)\frac{\lambda}{m^j}e^{-\frac{\lambda}{m^j}t}dt \\ &= \int_0^T D\frac{\lambda}{m^j}e^{-\frac{\lambda}{m^j}t}dt - \int_0^T t\frac{\lambda}{m^j}e^{-\frac{\lambda}{m^j}t}dt \\ &= D(1 - e^{-\frac{\lambda}{m^j}T}) - \int_0^T (e^{-\frac{\lambda}{m^j}t} - e^{-\frac{\lambda}{m^j}T})dt \\ &= D(1 - e^{-\frac{\lambda}{m^j}T}) - (-\frac{m^j}{\lambda}e^{-\frac{\lambda}{m^j}t} - e^{-\frac{\lambda}{m^j}T}t)\Big|_{t=0}^{T} \\ &= D(1 - e^{-\frac{\lambda}{m^j}T}) - \frac{m^j}{\lambda} + \frac{m^j}{\lambda}e^{-\frac{\lambda}{m^j}T} + Te^{-\frac{\lambda}{m^j}T}.\end{aligned}$$

Substituting $\mathbb{E}[\tau_j|\tau_j \neq D]$ into $\mathbb{E}[\tau_j]$ produces

$$\begin{aligned}\mathbb{E}[\tau_j] &= D\frac{1}{m^j} + \mathbb{E}[\tau_j|\tau_j \neq D]\frac{m^j - 1}{m^j} \\ &= D\frac{1}{m^j} - \left(\frac{m^j}{\lambda} - \frac{m^j}{\lambda}e^{-\frac{\lambda}{m^j}T} - Te^{-\frac{\lambda}{m^j}T} - D(1 - e^{-\frac{\lambda}{m^j}T})\right)\frac{m^j - 1}{m^j} \\ &= D\left(\frac{1}{m^j} + (1 - e^{-\frac{\lambda}{m^j}T})\frac{m^j - 1}{m^j}\right) - (1 - e^{-\frac{\lambda}{m^j}T})\frac{m^j - 1}{\lambda} + (\frac{m^j - 1}{m^j})Te^{-\frac{\lambda}{m^j}T} \\ &= D(1 - e^{-\frac{\lambda}{m^j}T}(1 - \frac{1}{m^j})) - (1 - e^{-\frac{\lambda}{m^j}T})\frac{m^j - 1}{\lambda} + (\frac{m^j - 1}{m^j})Te^{-\frac{\lambda}{m^j}T}.\end{aligned}$$

By summing over all the links in the tree we find the total multicast cost $C_{netw}^{multicast}$:

$$C_{netw}^{multicast} = b \cdot 60 \sum_{j=1}^{l} m^j \mathbb{E}[\tau_j]$$

and the average network bandwidth cost C_{netw}^{root} can be found by dividing by the average duration of each renewal cycle $(T + 1/\lambda) \cdot 60$.

$$C_{netw}^{root} = \frac{C_{netw}^{multicast} + C_{netw}^{unicast}}{(T + 1/\lambda) \cdot 60} = b \, \frac{\sum_{j=1}^{l} m^j \mathbb{E}[\tau_j] + lT^2\lambda/2}{T + 1/\lambda}.$$

Server costs for prefix server placed at the root:

It is easy to see that the storage cost C_{sto}^{root} of a single root server will be $b{\cdot}60D$. The I/O stream cost must be paid for the output capabilities of each server, i.e. the number of input/output streams which a server can simultaneously maintain. From the expression of C_{netw}^{root} above, one can conclude that the average number of streams for a root server is equivalent to

$$C_{I/O}^{root} = b \, \frac{D + T^2\lambda/2}{T + 1/\lambda}.$$

If T is chosen to minimize the number of I/O streams then

$$C_{I/O}^{root} = b(\sqrt{2D\lambda + 1} - 1).$$

However, choosing T to minimize the number of concurrent I/O streams will unnecessarily increase the network bandwidth. If T is chosen to minimize the network bandwidth, then $C_{I/O}^{root} = b(\sqrt{2CD\lambda/l + 1} - 1 - \frac{D\lambda(C/l-1)}{\sqrt{2CD\lambda/l+1}})$, where C is a scalar constant.

In case of a non-popular video that has on average, at most, one arrival in an interval of time D ($\lambda < 1/D$), each request activates a new prefix transmission and then a new cycle. Hence, the threshold time T becomes *zero* and the expressions for C_{netw}^{root} and $C_{I/O}^{root}$ simplify to

$$C_{netw}^{root} = bl\lambda D$$
$$C_{I/O}^{root} = b\lambda D \ .$$

Varying the placement of the prefix server We now generalize the model to the case where the prefix servers can be placed at any height in the network tree. By placing the prefix servers at some height h in the tree where $1 < h < l$, we divide the arrival process between m^{l-h} servers, each of which can be considered as the root of a network tree with height h. We need only therefore to consider the root server case for a tree of the proper height h with arrival rate λ/m^{l-h}, and then multiply the costs by the number of servers m^{l-h}. The resulting formulas are listed in table 1. The height $h = 1$ refers to the case of a leaf server, i.e. one level before the clients.

Cost terms	
C_{netw}^{prefix}	$b \cdot m^{l-h} \frac{\frac{hT^2\lambda}{2m^{l-h}} + \sum_{j=1}^{h} m^j \mathbb{E}[\tau_j]}{T + m^{l-h}/\lambda}$ $(\mathbb{E}[\tau_j] = D(1 - e^{-\frac{\lambda}{m^j m^{l-h}}T}(1 - \frac{1}{m^j})) - (1 - e^{-\frac{\lambda}{m^j m^{l-h}}T}) \frac{m^{l-h}(m^j-1)}{\lambda}$ $+ (\frac{m^j-1}{m^j}) T e^{-\frac{\lambda}{m^j m^{l-h}}T})$
C_{sto}^{prefix}	$b \cdot 60 \cdot m^{l-h} \cdot D$
$C_{I/O}^{prefix}$	$b \cdot \frac{D + T^2\lambda/2}{T + 1/\lambda}$

Table 1. Summary of the prefix cost terms for the PS-model (l levels, prefix servers at height h)

Costs for Suffix Transmission with a Multicast Tree The costs once again fall into three categories: bandwidth, storage capacity and streaming capacity. The bandwidth cost is equal to the transmission rate R_t^{min} multiplied by the average number of active links, which we will now calculate. For the periodic broadcast, each arrival is serviced for an amount of time $\mathbb{E}[T_s]$ (min), which equals the transmission time of the last segment:

$$\mathbb{E}[T_s] = \begin{cases} \lfloor \frac{L-D}{D} \rfloor \cdot D & \text{if } N_s > 1 \\ D & \text{if } N_s = 1 \end{cases}$$

We assume that all segments are multiplexed to a single multicast channel. As a consequence, each client will consume a bandwidth of R_t^{min} during all the transmission of the suffix. If one multicast channel were dedicated to each segment, the bandwidth consumption could be reduced; the client would be connected only to channels corresponding to segments not yet received. However, this reduction in bandwidth cost comes at the expense of a more complex multicast transmission and a complex synchronization between channels. This study is left for future work.

From queuing theory, it can be shown that given an expected service time $\mathbb{E}[T_s]$ and memoryless arrivals with parameter $\frac{\lambda}{m^j}$, the probability of n jobs simultaneously in progress is given by

$$\mathbb{P}\{n \text{ jobs}\} = \frac{e^{-\frac{\lambda}{m^j}\mathbb{E}[T_s]} (\frac{\lambda}{m^j}\mathbb{E}[T_s])^n}{n!},$$

which is a Poisson distribution. This result can be found through the derivation of the Erlang call-blocking formula commonly used in telecommunications. Arrivals to a link at level j are memoryless with parameter λ/m^j. Define $P(j)$ as the probability that a link at level j has any requests:

$$P(j) \triangleq 1 - \mathbb{P}\{0 \text{ jobs}\} = 1 - e^{-\frac{\lambda}{m^j}\mathbb{E}[T_s]}$$

Then

$$P(j) = \begin{cases} 1 - e^{-\frac{\lambda \lfloor \frac{L-D}{D} \rfloor D}{m^j}} & \text{if } N_s > 1 \\ 1 - e^{-\frac{\lambda D}{m^j}} & \text{if } N_s = 1 \end{cases}$$

The expected number of active links at any given time is therefore

$$\mathbb{E}[\text{active links}] = \sum_{j=1}^{l} m^j P(j),$$

and the bandwidth is

$$C_{netw}^{suffix} = R_t^{min} \sum_{j=1}^{l} m^j P(j). \tag{1}$$

On the other hand, the suffix is continuously and periodically multicast independent of the arrival rate as long as there is at least one user (if there are no users, the suffix server will not send the suffix). In contrast, the number of I/O channels does vary with L and D. The I/O stream cost is equal to the rate $R_t^{min} \times P(0)$, where $P(0)$ is the probability that there is at least one user active at the root:

$$C_{I/O}^{suffix} = \begin{cases} b \left(\sum_{i=1}^{\lfloor \frac{L-D}{D} \rfloor} \frac{1}{i} + \frac{\frac{L}{D} - \lfloor \frac{L}{D} \rfloor}{N_s - 1} \right)(1 - e^{-\lambda \lfloor \frac{L-D}{D} \rfloor D)}) & \text{if } N_s > 1 \\ b \, \frac{L-D}{D}(1 - e^{-\lambda D}) & \text{if } N_s = 1 \end{cases}$$

The storage cost is proportional to the length of the suffix, which is $C_{sto}^{suffix} = b \cdot 60(L - D)$.

The terms of the suffix costs are given in table 2, while table 3 shows all the important mathematical terms used.

Overall System Cost for the Case of a Single Video We divide the costs of a VoD service into network and server costs. The network costs are proportional to the amount of network bandwidth that is expended for the transmission of the prefix and the suffix. The server costs depend upon the necessary storage and upon the total number of input/output streams needed for the suffix server and the prefix server(s).

The total cost of the system can be computed as the sum of the total network and total server costs:

$$C_{PS}^{system} = C_{netw}^{system} + \gamma C_{server}^{system} \tag{2}$$

To relate the network and the server costs, a normalization factor γ is introduced that allows us to explore various scenarios for the cost of the servers as compared

Cost terms	
C_{netw}^{suffix}	$b\ (\sum_{i=1}^{\lfloor\frac{L-D}{D}\rfloor} \frac{1}{i} + \frac{\frac{L}{D} - \lfloor\frac{L}{D}\rfloor}{N_s - 1}) \sum_{j=1}^{l} m^j (1 - e^{-\frac{\lambda \lfloor\frac{L-D}{D}\rfloor D}{m^j}})$ if $N_s > 1$
	$b\ \frac{L-D}{D} \sum_{j=1}^{l} m^j (1 - e^{-\frac{\lambda D}{m^j}})$ if $N_s = 1$
C_{sto}^{suffix}	$b \cdot 60\ (L - D)$
$C_{I/O}^{suffix}$	$b\ (\sum_{i=1}^{\lfloor\frac{L-D}{D}\rfloor} \frac{1}{i} + \frac{\frac{L}{D} - \lfloor\frac{L}{D}\rfloor}{N_s - 1})(1 - e^{-\lambda \lfloor\frac{L-D}{D}\rfloor D})$ if $N_s > 1$
	$b\ \frac{L-D}{D}(1 - e^{-\lambda D})$ if $N_s = 1$

Table 2. Summary of the suffix cost terms for the PS-model.

Term	**Definition**
m	Tree breadth
l	Tree depth
h	Prefix server height
L	Video length (min)
D	Prefix length (min)
$L - D$	Suffix length (min)
N_s	Number of segments of the suffix ($N_s \triangleq \lceil\frac{L-D}{D}\rceil$)
λ	Average client request arrival rate (1/min)
N	Video popularity ($N \triangleq \lambda L$)
τ_j	Active occupation duration for link at depth j (prefix transmission)
$P(j)$	Prob. link at depth j is active (suffix transmission)
T	Threshold time
b	Consumption rate of the video [Mbit/sec]
C	Scalar constant

Table 3. Important Mathematical Terms

to the cost for the transmission bandwidth. We consider here the values of $\gamma = \{0, 0.1, 1\}$. The case of $\gamma = 0$ corresponds to the case that only the cost for network transmission is taken into account and the cost for the servers is not

considered at all (considered to be zero). $\gamma = 0.1$ provides high network cost relative to the server cost, while $\gamma = 1$ represents the case where the network cost is relatively low as compared to the server cost.

The terms for the network and server costs are given by:

$$C_{netw}^{system} = C_{netw}^{prefix} + C_{netw}^{suffix}$$
$$C_{server}^{system} = C_{server}^{prefix} + C_{server}^{suffix}$$

The server cost depends on both, the required amount of storage C_{sto} (in Megabit) and the amount of disk I/O bandwidth $C_{I/O}$ (in Megabit/sec).

$$C_{server}^{prefix} = \max(C_{I/O}^{prefix}, \beta C_{sto}^{prefix})$$
$$C_{server}^{suffix} = \max(C_{I/O}^{suffix}, \beta C_{sto}^{suffix})$$

To be able to relate the cost for storage and for I/O, we introduce the normalization factor β that is determined as follows: If our server has a storage capacity of d_{sto} [Megabit] and an I/O bandwidth of $d_{I/O}$ [Megabit/sec], then $\beta \triangleq \frac{d_{I/O}}{d_{sto}}$. Since the server will be either I/O limited (I/O is the bottleneck and no more requests can be served) or storage limited (storage volume is the bottleneck and no more data can be stored), the server cost is given as the *maximum* of $C_{I/O}$ and βC_{sto}.

To model a case where the cost for the "last-hop link" towards the clients is not the same as the cost for the other links, we can set the cost for the last link to the clients (lhc) to a value different from the cost for the other links.

2.4 Results for Long Videos

Introduction We consider a distribution network with an out-degree $m = 4$ and a number of levels $l = 5$. We expect that such a topology is representative for a wide distribution system that covers a large geographical areas of the size of a country such as France or the UK. If one wants to model a densely populated metropolitan area such as NewYork, one would choose $l < 5$ (e.g. $l = 2, 3$) and $m > 4$ (e.g. $m = 10$). Our model has quite a few parameters and we present results only for a limited subset of parameter values that can provide new insights. For the rest of the paper, we will vary only the parameters γ, last-hop cost lhc, and video length L. The other parameters are chosen as follows: For the disk I/O cost to disk storage cost ratio β, we choose $\beta = 0.001$, which is a realistic value for the current disk technology such as the IBM Ultrastar 72ZX disk. The video length will be $L = 90$ min for most of the time, except when we consider very short video clips of lengths $L = 2$ and 7 min.

Homogeneous Link Costs For the first results presented, the server cost is weighted by $\gamma = 1$ and the network per-link costs are uniform at all levels of the network ($lhc = 1$). We first plot in figure (see figure 6) the optimal system cost as a function of the video popularity N. The video popularity N represents the

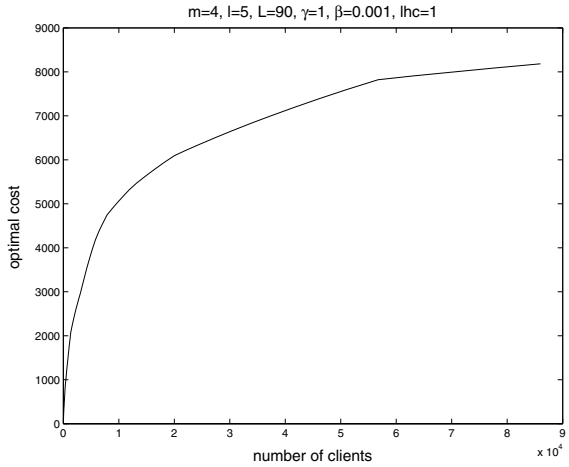


Fig. 6. Total system cost C^{system} with optimal prefix server height and optimal prefix length, for $\gamma = 1$ and $lhc = 1$

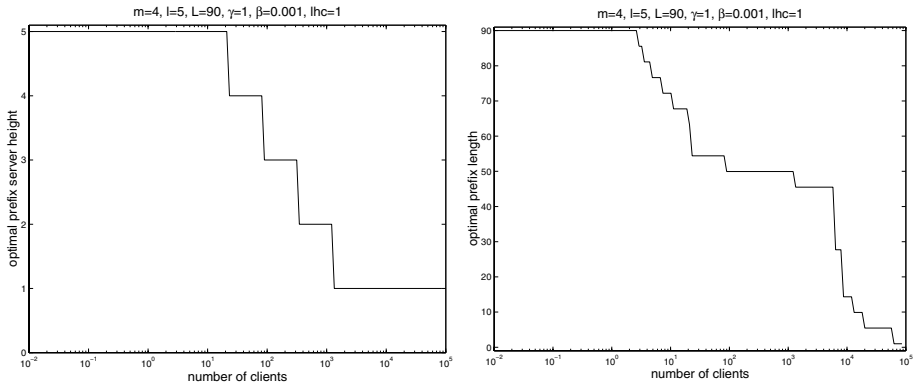
number of clients requesting a same video in an interval of time of duration L ($N = \lambda L$).

In figure 6 we see that the total cost efficiency improves with increasing video popularity N: For a 10-fold increase in video popularity N from 9,000 to 90,000 clients, the cost only doubles.

The optimal values for the prefix length and prefix server placement in the hierarchy as a function of the video popularity N are given in figures 7(a) and 7(b). We see that both, the prefix server height and the prefix length decrease monotonically with N.

For videos that are rarely demanded ($N < 20$), the prefix server is placed at the *root* and the optimal prefix comprises the whole video of 90 min. Indeed, for $N < 20$ the storage cost due to a replication of the prefix in multiple prefix servers is not justified and the optimal architecture is a *centralized* one. On the other hand, for videos that are popular or very popular, the optimal architecture is a *distributed* one with the server for the suffix at the root and the prefix servers closer to the clients. As the popularity N increases, the optimal prefix length decreases since the transmission bandwidth required by the prefix server increases with the square root of the number of clients served, while for the suffix server, the transmission bandwidth required depends for very high values of N only on the length of the suffix and not the number of clients served.

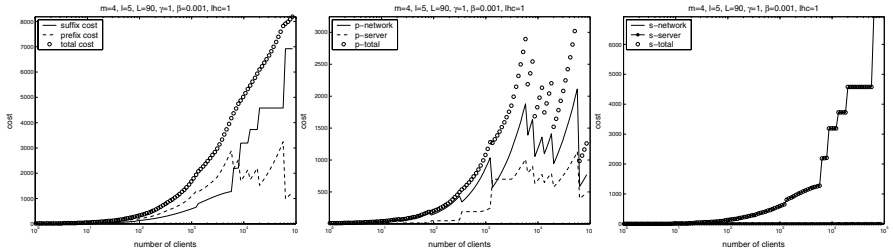
We plot the prefix-suffix cost breakdown in figure 8. We see that the suffix cost C^{suffix} is initially lower than the prefix cost C^{prefix} since the prefix length is quite long. Eventually, for an increasing video popularity N, the suffix system becomes more cost efficient, and so the length of the prefix is significantly reduced and the suffix cost becomes greater than the prefix cost. From figure 8(c) we see that the suffix cost C^{suffix} for higher values of N is entirely determined by



(a) Optimal prefix server height (b) Optimal prefix length (min)

Fig. 7. Optimal prefix server height and optimal prefix length for $\gamma = 1, lhc = 1$.

the suffix *network* cost. In the following, we will not present the suffix cost breakdown anymore since it does not provide any additional insight. For a given suffix length, the fact that C^{suffix} does not change with N indicates that *all* the links of the video distribution network are active, i.e. the multicast transmission is in fact a broadcast to all leaf nodes.



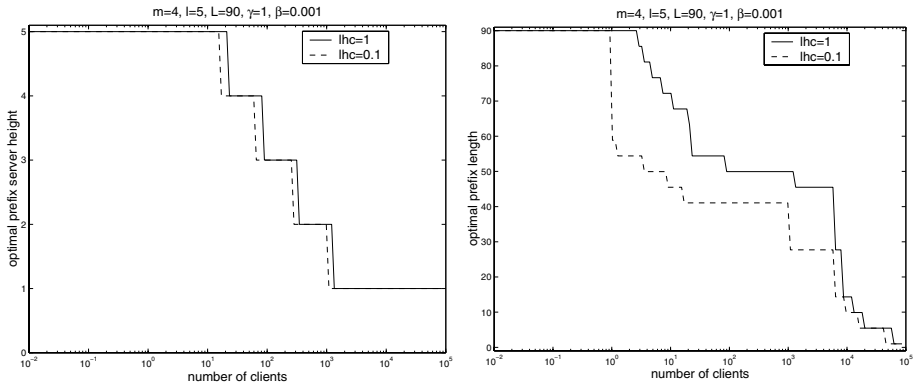
(a) total system cost (b) prefix cost breakdown (c) suffix cost breakdown

Fig. 8. Breakdown of costs for $\gamma = 1$ and $lhc = 1$

If we take a closer look at the evolution of the prefix *server* cost for very popular videos with $N > 10^3$, we see (figure 8(b)) that the prefix server cost increases linearly with increasing N. In this case, to achieve an optimal system cost C^{system}, the prefix length is frequently shortened.

Heterogeneous Link Costs We now set the cost of transmission between levels 4 and 5 (the "last-hop" from the root to the clients at the leaves) to be one-tenth the normal network cost. A reduced last-hop link cost would apply to situations where the local service providers are much cheaper than their regional and national counterparts since they support less traffic. A recent study [8] contains a cost comparison for transmission links of different bandwidth that indicates that the cost in Mbit/hour for local access via ADSL or Cable is at least one order of magnitude lower than the cost for a high speed OC-3 or OC-48 link. We can model this case by using a reduced last-hop link cost, which is $\frac{1}{10}$ of the cost for the other links. We refer to this case as $lhc = 0.1$.

In figure 9, we plot the optimal prefix lengths and prefix server heights under this scenario, with $m = 4, l = 5, \gamma = 1$ and $lhc = \{1, 0.1\}$.



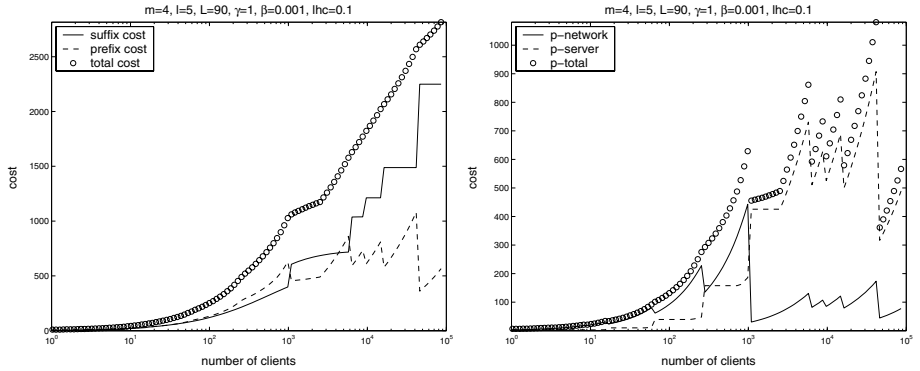
(a) optimal prefix server height (b) optimal prefix length

Fig. 9. Optimal prefix server height and optimal prefix length for $\gamma = 1$ and $lhc = \{1, 0.1\}$

Reducing the last-hop cost makes network transmission cheaper compared to server cost. We see that the prefix server heights are roughly the same while the prefix lengths decay faster than in the original setup ($lhc = 1$) to compensate for the relative increase in prefix server costs. This may seem counter intuitive since reducing the last-hop cost should make the prefix cost cheaper, especially if the prefix servers are at the leaves while the suffix server is at the root. However, we see in figure 10(b) that the *server* cost for the prefix C^{prefix}_{server} now dominates the total prefix cost C^{prefix}, (in particular when the prefix servers are placed at the leaves, i.e. for $N > 10^3$) which was not the case for $lhc = 1$ (see figure 8(b)).

When we compare the suffix costs in figures 8 and 10, we note that reducing the last-hop cost reduces the suffix network cost by almost a factor of 4, which assures that the suffix system remains cost effective. This cost reduction is due

to the fact that with $m = 4, l = 5$, there are 1024 links at the leaf level (last-hop) and only 340 links in the rest of the network. The network cost for the suffix system and the server cost for the prefix system are roughly in the same neighborhood. As a result, the server cost (i.e. the prefix server cost) is magnified in importance and the optimal policy is to compensate by shortening the prefix.



(a) total system cost (b) prefix cost breakdown

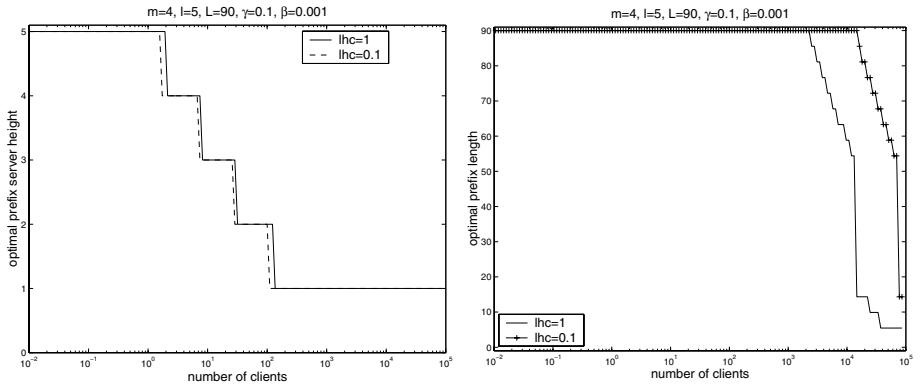
Fig. 10. Breakdown of costs for $\gamma = 1$ and $lhc = 0.1$

Reducing the Server Costs Relative to the Network Transmission Cost While the cost for servers is usually comparable in Europe and the US, the cost of network transmission is much *higher* in Europe than in the US. To account for the case where the network transmission cost relative to the server cost is higher, we set γ to 0.1.

The results for $\gamma = 0.1$ (figure 11) indicate that reduced relative server cost allows to deploy more servers in order to reduce the impact of the expensive network transmission cost. If we compare with $\gamma = 1$ (figure 9), we see that for $\gamma = 0.1$

- The optimal prefix (figure 11(b)) comprises the entire video for a much wider range of clients N. Only for a very high video popularity $N > 10^4$, the optimal prefix length decreases significantly.
- The prefix server height (figure 11(a)) drops faster to $h = 1$ since this helps to reduce the network costs and since the server costs, though they increase (the number of prefix server increases), have been discounted.

We also examine the case where $\gamma = 0.1$ and in addition the network cost for the last-hop is reduced to $lhc = 0.1$. We plot the optimal prefix lengths and prefix



(a) optimal prefix server height (b) optimal prefix length

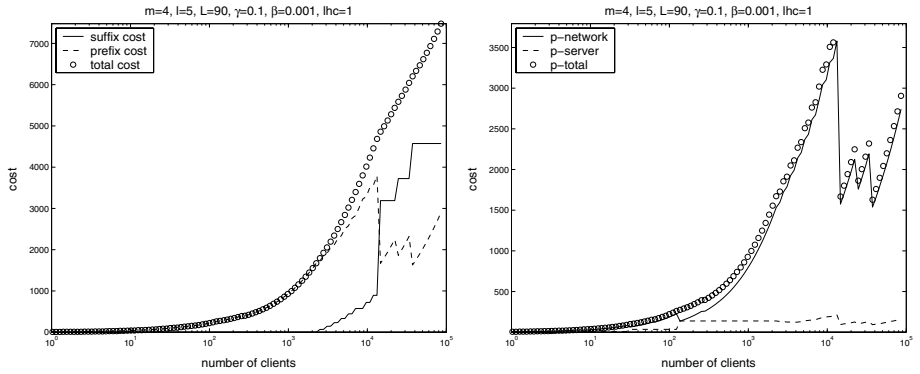
Fig. 11. Optimal prefix server height and optimal prefix length for $\gamma = 0.1$

server positions in figure 11. Previously, we saw for $\gamma = 1, lhc = 0.1$ that although the reduced last-hop cost significantly reduced the cost of prefix service (and of suffix service), the high server costs associated with leaf servers limited the use of the prefix. Now the server cost has been discounted, we see that reducing the last-hop network cost will allow the prefix servers to deliver the entire video over a wider range of video popularities as compared to the $lhc = 1$ case (figure 11). This is interesting, as reducing the last-hop network cost *decreased* the prefix length in the $\gamma = 1$ case and is an example of the interplay between network and server costs. For $\gamma = 1, lhc = 0.1$ the prefix server cost dominates the overall prefix cost (see figure 10(b), while for $\gamma = 0.1, lhc = 0.1$ it is the prefix network cost that dominates the overall prefix cost (see figure 13(b)).

Ignoring Server Costs Previous studies of video distribution schemes [13] have often ignored the server cost and only considered the network cost. We can model this case if we choose $\gamma = 0$. When we ignore the server cost, placing the prefix servers at the leaves is always optimal since the prefix *network* cost is minimized in this case.

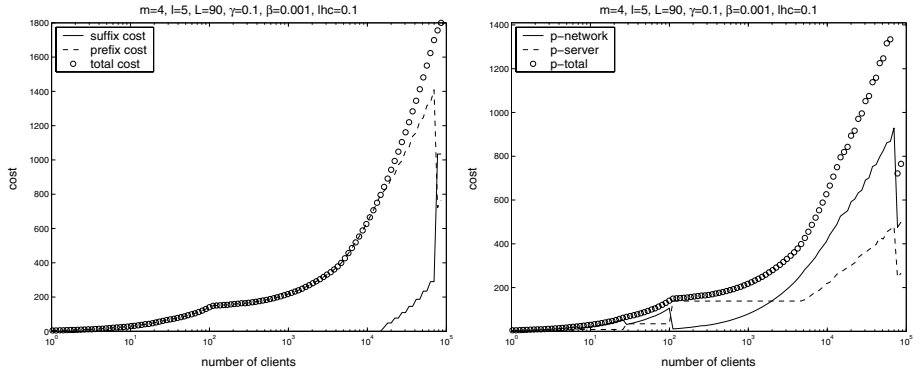
We plot the optimal prefix length for $\gamma = 0$ in figure 14(b). For both values of $lhc = \{1, 0.1\}$, the optimal prefix comprises the entire video for a large interval of the values of N. For large $N > 10^4$, the optimal prefix length becomes shorter as the centralized suffix system becomes more bandwidth efficient (despite the fact that the video must traverse 5 hops for the suffix as compared to 1 hop for the prefix) than the prefix transmission via controlled multicast.

If we compare the optimal values for the prefix length with the case of uniform link costs (i.e. $lhc = 1$) and large N ($N > 10^4$) , we see that for $\gamma = 0$ the optimal values are only unnoticeably larger than for the case of $\gamma = 0.1$.



(a) total system cost (b) prefix cost breakdown

Fig. 12. Breakdown of costs for $\gamma = 0.1$ and $lhc = 1$



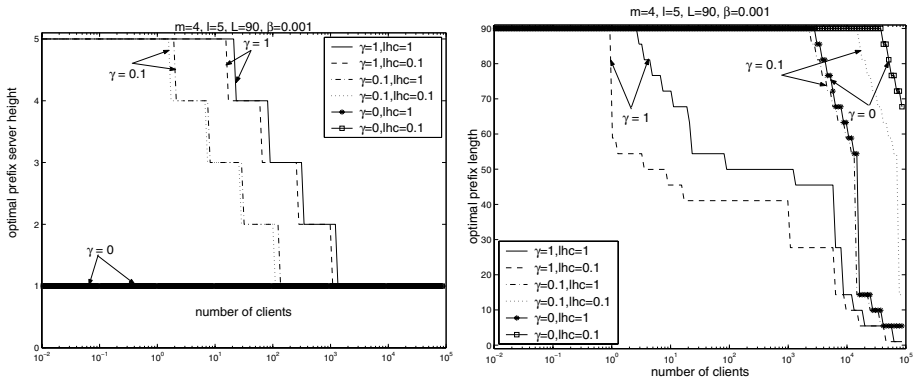
(a) total system cost (b) prefix cost breakdown

Fig. 13. Breakdown of costs for $\gamma = 0.1$ and $lhc = 0.1$

We plot the optimal prefix server height and optimal prefix lengths for all configurations covered so far in figure 14. We see how the system adapts the partitioning into prefix and suffix and the placement of the prefix servers as a function of the cost for the network and server resources. When the cost for the servers relative to the cost for network transmission is reduced ($\gamma < 1$) more prefix servers are deployed. For a given video popularity N, this happens in two ways

- The prefix servers are placed closer to the clients (see figure 14(a))
- The prefix is made longer (see figure 14(b))

In the case of $\gamma = 0$, the entire video is, over a wide range of the video popularity N, delivered by the prefix servers.



(a) optimal prefix server height

(b) optimal prefix length

Fig. 14. Optimal prefix server height and optimal prefix length for $\gamma = \{1, 0.1, 0\}$ and $lhc = \{1, 0.1\}$

We conclude this subsection by showing in figure 15 the optimal total cost values under each scenario discussed. We see that

- for the topology $m = 4, l = 5$ chosen, the majority of the links are at the last-hop to the leaves. Therefore, for non-uniform link costs ($lhc = 0.1$) the cost of the overall system is considerably much smaller than for $lhc = 1$.
- There is surprisingly little difference in the cost of the overall system for $\gamma = 0.1$ and $\gamma = 0$ since in both cases it is the cost for the network transmission that dominates the total cost. For $\gamma = 0.1$, the impact of the prefix *server* cost is attenuated (for $1 < N < 100$) by using fewer prefix servers (see figure 14(a))

Conclusions So Far We have seen how our distribution architecture gives us the cost-optimal configuration as a function of the video popularity N. For a non-popular video, the prefix server is placed at the root[6] and the length of the prefix comprises the whole duration of the video. As the video popularity N increases, the optimal prefix length becomes shorter and the optimal height for prefix servers becomes closer to the clients.

[6] With the exception of $\gamma = 0$, where the prefix servers are always placed at height $h = 1$.

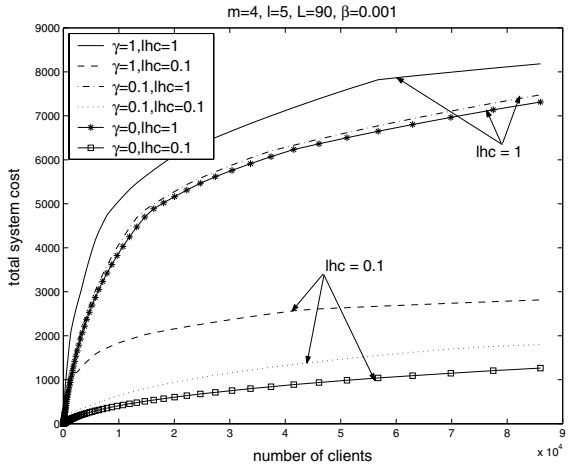


Fig. 15. Comparison of total system costs for $\gamma = \{1, 0.1, 0\}$ and $lhc = \{1, 0.1\}$

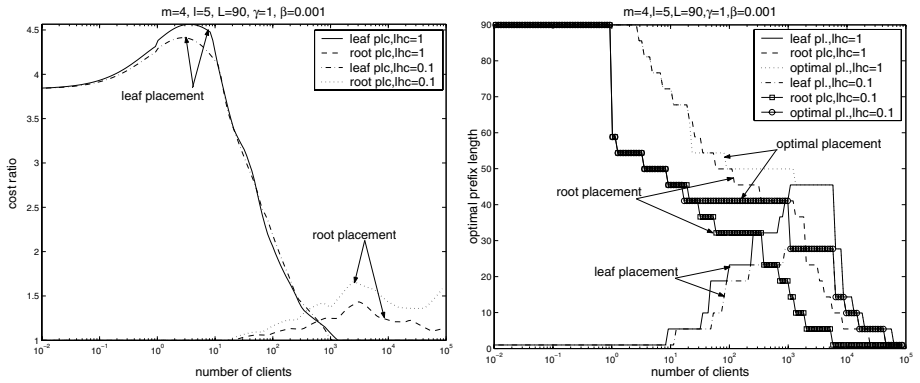
Using the suffix server at the root of the distribution tree becomes economically very interesting for very popular videos, where the prefix length becomes much shorter than the whole duration of the video. However, the optimal suffix length used is quite sensitive to changes in the popularity (see figure 14(b) for $10^4 \leq N \leq 10^5$). Therefore, to operate the distribution system in a cost-optimal fashion, one needs to periodically estimate the current popularity of a video and adapt, if necessary, the prefix length.

Costs for Non-optimal Prefix Server Placement For a large video distribution system serving many movies, it will be feasible to place prefix servers at every level of the distribution tree; however, for smaller systems, the prefix server locations will probably be fixed as there may not be prefix servers at every level of the distribution network. The placement of the prefix servers at an arbitrary level may also be impossible for other reasons. For instance, the facilities necessary for installing a server may not be available or accessible. Thereby, it is worth evaluating the cost performance of a video distribution system where the prefix servers are placed non-optimally. We will examine how the system adjusts the prefix length to find the most cost-efficient solution for the case where

- The prefix server is placed at the root, which will be referred to as **root placement**
- The prefix servers are placed at the leaves (i.e. at height h = 1, one level above the clients), which will be referred to as **leaf placement**

We always assume that the cost for the server is taken into account (i.e. $\gamma \neq 0$) and consider the following scenarios

- Network is cheap, $\gamma = 1$ and $lhc = \{1, 0.1\}$.
- Network is expensive, $\gamma = 0.1$ and $lhc = \{1, 0.1\}$.



(a) Cost ratio for prefix server, $\gamma = 1$, $lhc = \{1, 0.1\}$

(b) Optimal prefix length, $\gamma = 1$, $lhc = \{1, 0.1\}$

Fig. 16. System cost ratio for non-optimal placement to optimal placement system cost and optimal prefix length for $\gamma = 1$ and $lhc = \{1, 0.1\}$.

Network is cheap We first discuss the case with $lhc = 1$, where the per-link network costs are the same across the network. We know from figure 7(a) that for values of $N, N < 20$, the optimal placement of the prefix server is at the root. For increasing values of $N > 20$, the optimal placement finds the best trade-off between network and server cost by moving the prefix servers closer to clients and reducing the length of prefix. When we fix the placement of the prefix server at the root, the only degree of freedom left to minimize the cost is to adjust the prefix length. We see in figure 16(b) that the prefix length for the root placement case decreases more rapidly than when the prefix placement is chosen optimally.

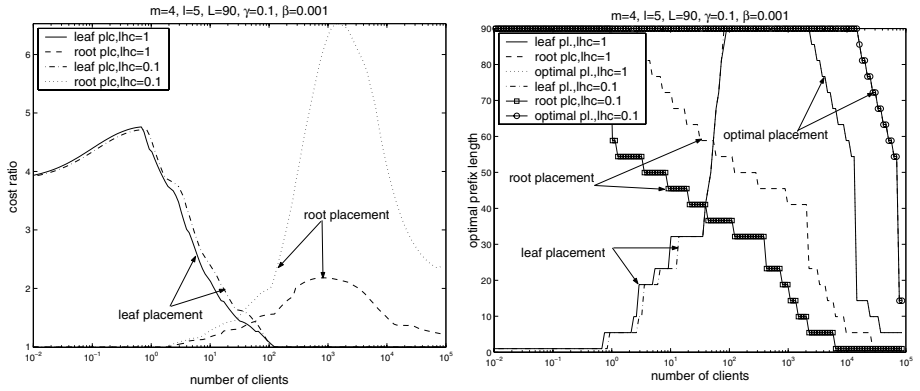
In figure 16(a) we plot the ratio of the total system costs. For the case when the prefix server is always at the root as compared to when it is optimally placed, the total system cost increases by up to 60%: For very popular videos, placing the prefix server at the root results in a longer suffix in order to limit the high cost for the prefix distribution.

When the prefix servers are always at the leaves, the cost increase as compared to the optimal placement is much higher than for root placement and can be up to 450% since leaf placement is very expensive for non-popular videos ($N < 10^2$). This big difference in cost is due to the fact that keeping copies of a video in all the prefix servers placed at the leaves is very inefficient for videos that are not very popular, i.e. $N < 10^3$. When the prefix servers are placed at the leaves and the video popularity is low ($N < 10^1$), the system chooses the minimal possible prefix length of $D = 1$ to limit the overall cost of keeping the many copies of that prefix (see figure 16(b)). For larger values of N, the optimal prefix server placement is at the leaves, and so the performance of the non-optimal system is the same as the optimal one for $N > 10^3$.

When we compare the case where all links of the network have the same cost ($lhc = 1$) to the case where the last-hop links are less expensive for $lhc = 0.1$, we see little difference in the results. For $lhc = 0.1$, the worst case cost increase due to leaf placement is a little bit less than for $lhc = 1$. For the root placement, the fact that the last-hop links are less expensive ($lhc = 0.1$) increases the cost of root placement as compared to $lhc = 1$.

Overall, when the network is cheap and we must choose between root and leaf placement, root placement is preferable.

Network is expensive We examine now the opposite situation, where the network is expensive as compared to the cost for servers (see figure 17(a)).



(a) cost ratio for prefix server, $\gamma = 0.1$, $lhc = \{1, 0.1\}$

(b) Optimal prefix length, $\gamma = 0.1$, $lhc = \{1, 0.1\}$

Fig. 17. System cost ratio for non-optimal placement to optimal placement and optimal prefix length for $\gamma = 0.1$ and $lhc = \{1, 0.1\}$

When the network is expensive, the worst case cost performance of leaf placement deteriorates only slightly as compared to the case when the network is cheap, since the cost for the servers placed at the leaves dominates the total system cost. For root placement, the worst case cost is significantly higher as in the case of $\gamma = 0.1$, in particular for $lhc = 0.1$, since the network transmission has become more expensive, which is directly reflected in an increase of the total system cost. The optimal placement for $\gamma = 0.1$ moves the prefix servers for increasing N rapidly towards the clients and chooses a prefix that comprises the entire video for all except the very popular ones $N > 10^4$ (see figure 14). Since the root placement can not move the prefix servers, it shortens the prefix length drastically with increasing N to offset the cost-increase due to the prefix placement at the root (see figure 17(b)).

Conclusion The video delivery architecture has normally two degrees of freedom: the prefix length and the prefix server placement can be varied to determine the cost optimal solution. When we remove one degree of freedom and fix the placement of the prefix server, the total system cost can increase significantly, in particular for the case of leaf placement, where the server cost dominates as compared to the network cost.

2.5 Short Videos

Introduction So far we have looked at videos of length $L = 90$ min, which corresponds to feature movies. Besides movies, there are news clips or clips for product promotion, which are much shorter in length, that can be distributed via a video distribution system. For these clips it is interesting to evaluate how efficiently the video distribution architecture supports the distribution of these clips. We consider clips of $L = 7$ min[7]. As before, the network has an outdegree $m = 4$ and a number of levels $l = 5$. We vary the popularity of the video between $10^{-2} \leq N \leq 10^5$.

The optimal configuration is computed using the PS-model. The PS-model allows to determine which fraction of the video should be stored at the prefix servers and where to place the prefix servers to minimize the delivery cost.

In addition, we consider two more cases where we remove one degree of freedom:

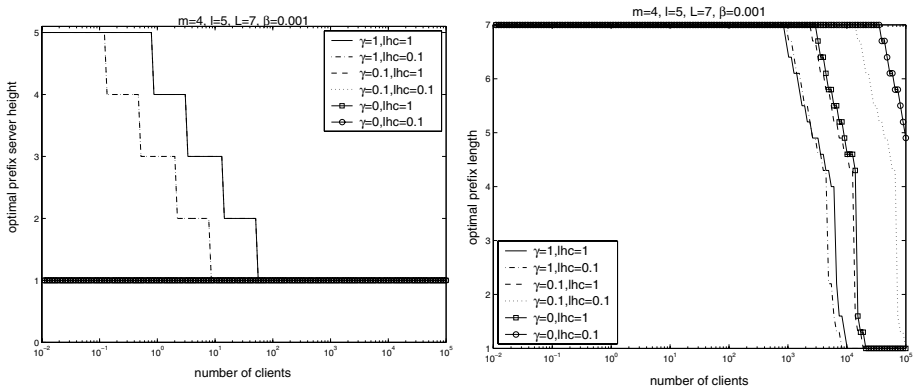
- $D = L$, i.e. the prefix comprises the full video, which we refer to as **full video caching**. However, the height of the prefix servers is chosen such to minimize the overall system cost.
- The prefix server is fixed at the root, which we introduced in subsection 3 as **root placement**. However, the length of the prefix is chosen such to minimize the overall system cost.

Results Figure 18 shows the optimal prefix server placement and the optimal prefix length in the hierarchy as a function of the video popularity N. A comparison with the values obtained for long videos of $L = 90$ min (see figure 14) shows that the video distribution system behaves the same way, for both short and long videos:

- With increasing video popularity N, the placement of the prefix servers moves closer to the clients
- For all but the most popular videos, the optimal prefix comprises the whole video.

As we can observe from figures 18(b) and 19(a), full video caching is the optimal solution, except for the most popular videos.

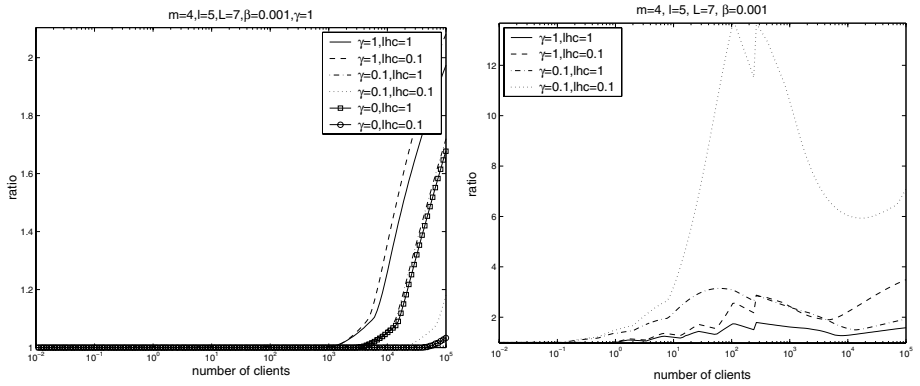
[7] We also looked at clips of 2 min length. The results we have obtained for $L = 2$ are very similar to the ones for $L = 7$ and are therefore not present here.



(a) optimal prefix server height

(b) optimal prefix length

Fig. 18. Optimal prefix server height and prefix length for $\gamma = \{1, 0.1, 0\}$, $lhc = \{1, 0.1\}$, and $L = 7$ min.



(a) Cost ratio for full video caching

(b) Cost ratio for root placement

Fig. 19. Cost ratio for full video caching system and cost ratio of root placement for $\gamma = \{1, 0.1, 0\}$, $lhc = \{1, 0.1\}$, and $L = 7$ min.

In Figure 19(b) we plot the ratio of the delivery cost of a video obtained in the case of root placement as compared to the delivery cost obtained when both, the prefix length and the prefix server placement are chosen optimally[8]. We see

[8] We do not plot the cost ratio for $\gamma = 0$, which can reach a value up to 40 for small values of N.

that there is an additional cost of fixing the prefix server at the root for a values of video popularity N except very small ones, where placing the prefix server at the root is the optimal choice. The additional cost due to root placement is lowest when the network transmission is cheap ($\gamma = 1$) and highest when the relative cost for the prefix servers is low ($\gamma = 0.1$) **and** the transmission cost over the last hop is reduced ($lhc = 0.1$). When the network is expensive ($\gamma = 0.1$) the cost ratio is worst for the values of N where the optimal prefix server placement puts the prefix servers at the leaves ($h = 1$) and chooses a prefix that comprises the entire video ($D = L$). The shape of the curves is similar to the ones observed for long videos (see figures 16(a) and 17(a)).

2.6 Video Distribution System for a Set of Videos

Introduction So far we have looked at a single video. We studied the case of how to determine the optimal prefix length and the optimal prefix placement as a function of the video popularity N. However, a complete video distribution system will offer to the clients a host of different videos $\mathcal{V} = \{1, \ldots, K\}$ to choose from.

We will now extend the PS-model to deal with the following scenarios:

- Provisioning.
  The PS-model is used to solve the provisioning problem for a given set of videos whose popularities are known: We just need to execute the model for each video separately to determine the optimal prefix length and the placement of the prefix servers.
- Video assignment problem for an existing configuration.
  Very often, the situation will be such that the video distribution system has been already deployed, i.e. the central suffix server and the prefix servers have been installed and changing the location (placement) or the capacities of prefix servers is not possible. Given that the placement and the capabilities of the prefix servers are *fixed*, we then want to determine the cost-optimal prefix length and prefix server placement for a set of videos and popularities.

For a set $\mathcal{V} = \{1, \ldots, K\}$ of K videos, the PS-model computes separately the optimal system cost of each video $i \in \mathcal{V}$. The total system cost will be the sum of the system costs over all K videos of the system. In the following, we will consider that the popularity $N_i = \frac{A}{i^\alpha}$ of video $i \in \mathcal{V}$ follows a Zipf distribution the popularity of the most popular video and α the slope of the popularity distribution; the bigger α, steeper the slope, i.e. the more biased or skewed the popularity distribution. In figure 20 we plot the popularity for different values of A and α.

Provisioning of the Prefix Servers The aim of provisioning is to compute the resources required in terms of video server storage and video server I/O bandwidth for a given set of videos $\mathcal{V} = \{1, \ldots, K\}$ with popularities N_i, for

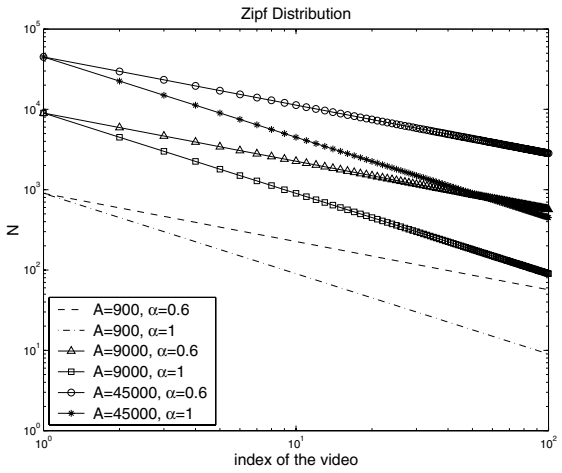


Fig. 20. The popularity N_i of video i as a function of the index i, according to a Zipf distribution, on a log-log scale.

$i \in \mathcal{V}$[9]. We will concentrate here on the provisioning of the prefix servers inside the network. However, the provisioning of the servers at the root can be done in a similar way.

We use equation 2 (subsection 2.3) to compute the optimal prefix length D_i, the optimal threshold T_i, and the optimal prefix server level l_i for each video $i \in \mathcal{V}$. Let $\mathcal{L}(j) \triangleq \{i \in \mathcal{V} \mid l_i = j\}$ denote the subset of videos whose prefix will be optimally served by the prefix servers of level j and I/O_i denote the amount of I/O bandwidth needed for each prefix server at level j to serve the prefix of video $i \in \mathcal{V}$. The value of I/O_i can be computed the same way as $C_{I/O}^{prefix}$ in subsection 2.3. With $\lambda_i = \frac{N_i}{L}$ and $i \in \mathcal{L}(j)$ we have

$$I/O_i = b\ \frac{\lambda_i}{m^j} \frac{2D_i + \lambda_i T_i^2/m^j}{2 + 2\lambda_i T_i/m^j}$$

At each level j, the total storage $PS_{st}(j)$ and I/O $PS_{I/O}(j)$ capacity of the prefix servers are computed as the sum of the prefix lengths and the amount of I/O bandwidth needed over all prefixes placed at that level. Hence, at level j, the resource requirements of the prefix servers are given by:

$$PS_{st}(j) = \sum_{i\ \in\ \mathcal{L}(j)} b \cdot 60\ D_i \qquad \forall j \in \{1, \ldots, l-1\}$$

$$PS_{I/O}(j) = \sum_{i\ \in\ \mathcal{L}(j)} I/O_i \quad \forall j \in \{1, \ldots, l-1\}$$

[9] For sake of simplicity we assume that all videos have the same length L.

Since we assume a homogeneous client population, the load will be uniformly distributed over the prefix servers at a particular level. Therefore, all prefix servers at a particular level j will have the same capabilities.

Assignment of a Videos to Prefix Servers with Limited Storage and I/O Capabilities We now consider the case that the prefix servers have been installed and that it is not possible to add new prefix servers or to modify their placement in the distribution hierarchy. The values of $PS_{st}(j)$ and $PS_{I/O}(j)$ are known $\forall j \in \{1, \ldots, l-1\}$. We will refer to them as **prefix server constraints**. However, we allow for modifications to the servers installed at the *root*. This means that there are no constraints on the resources of the central suffix server or the prefix server at the root ($l = 0$).

To solve the prefix assignment problem for a set of videos $\mathcal{V}$, we need to find the placement that satisfies the prefix server constraints of the system and minimizes the total system cost.

We formulate the assignment problem for a set $\mathcal{V}$ of videos as follows:

$$\begin{cases} \min\limits_{\theta_{ij}}(\sum\limits_{j=0}^{l-1}\sum\limits_{i\in\mathcal{V}} C_{PS}^{system}(i,j) \times \theta_{ij}) & \\ s.t. \quad \sum\limits_{i\in\mathcal{V}} D_{ij} \times \theta_{ij} \leq PS_{st}(j) & 1 \leq j \leq l-1 \\ \quad \sum\limits_{i\in\mathcal{V}} I/O_{ij} \times \theta_{ij} \leq PS_{I/O}(j) & 1 \leq j \leq l-1 \\ \quad \sum\limits_{j=0}^{l-1} \theta_{ij} = 1, & \forall\ i \\ \quad \theta_{ij} \in \{0,1\}, & \forall\ i,j \end{cases}$$

where (i) $C_{PS}^{system}(i,j)$ is the lowest total system cost achievable when placing video i at level j, (ii) D_{ij} is the corresponding optimal prefix length when placing video i at level j, (iii) I/O_{ij} is the amount of I/O bandwidth needed when placing video i at level j, and (iv) θ_{ij} is a binary variable. θ_{ij} is equal to 1 if the prefix of the video i is placed at level j and 0 otherwise. $\sum\limits_{j=0}^{l-1} \theta_{ij} = 1$ indicates that no video prefix can be stored at more than one level in the hierarchy.

Both, the objective function and the constraints are linear functions of the binary variables θ_{ij}. This optimization problem can be resolved using dynamic programming. We use the *XPress-MP* package [15] to solve the assignment problem.

Results Moving a video prefix from an optimal level to a non-optimal one in order to satisfy the constraints increases the delivery cost of the video and consequently the overall system cost. It is interesting to evaluate how the prefix

server constraints will impact the overall system cost of the video distribution system. To this purpose, we compute the overall system cost C_{opt}^{vds} *without* any constraints and the overall system cost C_{constr}^{vds} with prefix server constraints, which are defined as

$$C_{opt}^{vds} = \sum_{i \in \mathcal{V}} C_{PS}^{system}(i)$$

$$C_{constr}^{vds} = \sum_{j=0}^{l-1} \sum_{i \in \mathcal{V}} C_{PS}^{system}(i,j) \times \theta_{ij}$$

We use the cost ratio $C_{constr}^{vds}/C_{opt}^{vds}$ to evaluate how well the video distribution architecture can adapt to changes in the video popularity and the number of videos. We plot in figure 21 the cost ratio for $\alpha = \{0.2, 0.6, 1\}$ and different numbers of videos $K = \{100, 120, 150\}$. We set the normalization constant to $A = 9000$. The prefix server constraints were computed for the case of $K = 100$ videos with $\alpha = 0.6$ for the Zipf distribution. This initial distribution ($\alpha = 0.6$) is skewed or biased enough to fit well with small systems.

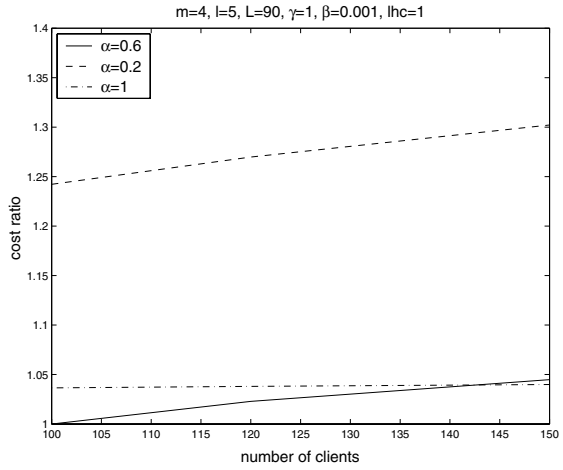


Fig. 21. Cost ratio $C_{constr}^{vds}/C_{opt}^{vds}$.

Figure 21 shows the following interesting features:

- The cost ratio increases sub-linearly with the number of videos.
- The increase of the system cost strongly depends on the parameter α.

These results fit well with the intuition. The larger the popularity N of a video, the closer is the placement of the prefix to the clients. If there is no more place for the popular videos at the higher levels of the prefix hierarchy, video must be moved closer to the root.

For a given Zipf distribution, increasing the number of videos adds videos at the *tail* of the distribution. Those additional videos are the least popular ones, and must be moved near the root to free resources for the more popular ones at the higher levels. The placement of the least popular videos close to the root will use up very few of the constraint resources, which explains the small change in the cost ratio with increasing number of videos. At some point, all the additional videos will have such a low popularity that their prefixes will all be placed at the root and the cost ratio will no longer change with increasing number of videos.

For $\alpha = 1$, the popularity distribution is very skewed, and increasing the number of videos K in the system has no impact on the cost ratio since those additional videos are all optimally placed at the root. If we compare $\alpha = 0.2$ and $\alpha = 0.6$, we find that, as the number of videos K increases, the cost ratio increases more rapidly for $\alpha = 0.2$ than for $\alpha = 0.6$. The reason is that, for $\alpha = 0.2$, the videos added to the system are relatively more popular than those for $\alpha = 0.6$, and as a consequence, moving them closer to (or placing them at) the root to satisfy the constraints comes out more costly.

We also evaluated other scenarios such that, $\gamma = 0.1$, or $\gamma = 0$ and for a reduced cost of the last hop link $lhc = 0.1$. The results obtained were similar and are therefore not presented here.

Conclusion The PS-model adapts itself very well in case of limited prefix server resources. Non-popular videos are moved close to the root to free a place for the more popular ones. For $\alpha = 0.6$ and $\alpha = 1$, the increase in the overall system cost for increasing number of videos is very low, less than 5%. Even when the popularity distribution differs a lot from the one used when determining the prefix server resources (as is the case for $\alpha = 0.2$) the extra cost incurred will be small, 25%– 30%.

2.7 Conclusion and Outlook

Summary We have presented a scalable video distribution architecture that combines open-loop and closed-loop schemes and assure a zero start-up delay. Under this architecture, each video is split into two parts, the prefix and the suffix. The prefix is transmitted via controlled multicast (closed-loop scheme) while the suffix is transmitted via tailored periodic broadcast (open-loop scheme). The architecture is very cost-effective since the cost for the prefix transmission increases only with the square root of the number of clients and the suffix distribution cost is, at high request rates, independent of the number of clients and simply a function of the number of segments the suffix is decomposed into.

Another advantage is that our architecture is highly scalable in terms of serving incoming client requests. A client who wants to receive a video contacts his closest prefix server. The central server only needs to know if there is at least one client connected, which the central server can learn by communicating with the prefix servers. This interaction between the clients and the servers avoids having a bottleneck due to handling all the requests by a single server.

We have developed an analytical cost model for that architecture that we called PS-model. In the cost model, we include not only the network bandwidth cost, but also the costs for the server I/O bandwidth and server storage. Using the PS-model we can determine the

- Bandwidth and streaming costs for prefix and suffix transmissions
- Optimal prefix length
- Optimal position of the prefix servers.

The PS-model makes the trade-off between the server and network bandwidth costs to determine the cost-optimal prefix length and the optimal prefix server placement.

As key results, we found that

- The cost efficiency increases with increasing video popularity N. For a 10-fold increase in N from 9,000 to 90,000 clients the total system cost only doubles.
- A popular video is replicated at many prefix servers that are placed close to the clients.
- A non-popular video is placed at the root or very close to the root to reduce the server storage cost incurred for replicating the prefix across many prefix servers.
- The central suffix server is highly efficient to serve very popular videos for which the optimal the suffix comprises the major portion of the video.

The PS-model has two degrees of freedom: It can adjust the length of the prefix as well as the placement of the prefix server in the hierarchy so as to divide efficiently the workload between the suffix server and the prefix servers. Thus, if we remove one degree of freedom, for instance, we fix the placement of the prefix server at either the root or the leaves, the only degree of freedom left is to adjust the length of the prefix. It is worth to discuss the cost for a non-optimal placement of the prefix servers since for small systems, it might not possible to have prefix servers at any level in the network. By comparing between leaf placement and root placement for the prefix servers, we found that

- The system cost can increase significantly as compared to the optimal system cost. This increase is up to 450% for leaf placement.
- Root placement outperforms leaf placement in the case where the network cost is cheap as compared to the server cost. In this case, the cost increase is relatively small, up to 60%.
- Root placement becomes very costly when the network cost is high relative to the server cost, and the increase in the system cost can exceed 600% in case of a reduced last-hop cost.

We also evaluated the PS-model for the case of short videos such as video clips or clips for product promotions. In fact, even for short videos, it is always cost-optimal to divide the video into prefix and suffix in the case very popular clips.

Moreover, we extended the PS-model to looked at the following scenarios:

- Provisioning:
  We showed how the PS-model can be used to compute the cost-optimal for a system with a set $\mathcal{V}$ of videos. Given the popularity of each video, the PS-model determines the system resources required in terms of network bandwidth, server I/O bandwidth, and server storage to optimize the total system cost.
- Assignment of prefixes into prefix servers:
  We studied how the PS-model adapts the prefix length and the prefix placement for a set $\mathcal{V}$ of videos when the amount of resources for the prefix servers is given, which is for instance the case when a new set of videos must be optimally placed. The cost increase due to predefined capabilities of the prefix servers is very low over a wide region of our model, less than 5%. Even when the popularity distribution of the videos differs a lot from the initial distribution used to determine the prefix server resources, the increase in the system cost remains low, 25-30%.

The PS-model has allowed us to study different scenarios. However, several extensions to the PS-model are possible that allow to make the overall model more realistic. The current model assumes that client requests for a single video are homogeneously distributed among all clients. A possible extension would consider the case where a particular video is more popular with a sub-group of the clients which would lead to heterogeneous request patterns. The current video distribution network has a very regular structure with all clients being at the same distance from the root and the distribution tree being very regular, while a real distribution network most likely has not such a regular structure. We intend to extend our model in the future and evaluate the impact of these extensions. These extensions will clearly change the absolute cost values. However, we do not expect that they will change the broad conclusions that we could draw using the PS-model.

We would also like to evaluate different architectural choices. Today, digital VCRs with at least one hundred Gigabyte of local storage are commercially available [40]. Given local storage, one can proactively download the prefixes of the most popular videos directly into the VCR. We intend to extend the PS-model to evaluate the overall cost reduction due to the use of local storage in the VCR. Another very attractive architectural option is a satellite distribution of the suffix of the videos.

Outlook In the system model we have analyzed, we had assumed that both, the suffix server and the prefix servers, are *dedicated*. These assumptions hold true for a "commercial" CDN. In recent years, a new paradigm called peer-to-peer [34] emerged where a particular machine can assume both roles, i.e. be client and server at the same time. Popular peer-to-peer systems such as Gnutella, Napster, or KaZaa have been used by Millions of users to share digital content such as MP3 files. P2P architectures are very interesting for scalable

content distribution. They offload all or at least the majority of the work for storage and distribution onto end-systems that are *not dedicated* to the purpose of content distribution and therefore, significantly reduce capital expenditures. P2P architectures are also inherently *self-scaling*: In case of a sudden increase in the number of requests, the number of peers that have received the content will also increase, which in turn will increase the total capacity of the P2P system to serve new clients. P2P systems are therefore much more suitable than centralized server-based systems to handle "flash crowds".

The research community has made numerous proposals on how to use the P2P paradigm to provide overlay multicast distribution trees [14, 7] and to perform scalable video distribution. The P^2Cast scheme [22] partitions the video into prefix and suffix as we do in our model and proposes to distribute the suffix by a central server via application layer multicast while the prefix is delivered via unicast by another client that has already received the prefix previously and is currently viewing the suffix.

Constructing application layer multicast trees for video distribution is very challenging as clients that are part of the tree may leave at any time, which may disrupt the video reception of the clients that are downstream while the tree is re-built. Various proposals such as Coopnet [36] and SplitStream [12] propose to encode the video signal using multiple description encoding techniques [20] where the video signal is encoded as N independent descriptions or sub-streams that are each transmitted on a separate multicast tree. In this case, a receiver will be able to play out the video, albeit with reduced resolution, if it receives only a subset of the descriptions. When using such an approach, it is important that the multicast trees for transmitting the different descriptions are constructed in such a way that the failure of a node will not affect different receivers for each of the descriptions. This is assumed in SplitStream by constructing the trees in such a way that an interior node in one tree is a leaf node in all the other trees.

A hybrid architecture that combines a CDN (with its servers installed at certain places) and peer-to-peer based streaming was proposed in [47]. Such an architecture allows to considerably reduce the amount of CDN resources required since peers that have received a video will in turn serve other clients, which reduces that load on the CDN server.

3 Scheduling Objects for Broadcast Systems

3.1 Introduction

Technological progress in high speed communications, embedded systems and consumer electronics has led to the availability of several *thin*, personalized user terminals capable to store, process, transmit and receive information, such as mobile phones, personal digital assistants (PDA), palmtops, tablet PCs, etc. Their powerful characteristics enable a wide range of services and applications, which deliver information to users efficiently.

Broadcast systems constitute a popular communication infrastructure to deliver services to thin, personalized *client* devices. Broadcast systems are capable

to deliver multimedia services to a wide client population, because they are scalable in terms of user population and end-user bandwidth; furthermore, they accommodate heterogeneous user technologies, including mobile and wireless [4]. There exist several architectures for broadcast systems differing in the data delivery methods and mechanisms, such as the push or pull data transmission model, periodic or aperiodic transmission, etc. Each architecture provides advantages for specific environments and/or set of services.

In this work, we consider a popular and powerful broadcast system, suitable for satellite systems: an asymmetric, push-based system with receive-only i clients (i.e. without uplink bandwidth), who are mobile, or in general, connected occasionally. Typical broadcast systems in this category include the deployed systems Pointcast [28], Traffic Information Systems [38], Stock, Weather and News dissemination systems, DirecPc by Hughes [16] and Internet-over-Satellite (IoS) by Intracom [29]. In analogy to a web-type environment, we assume that application information is composed of logical objects (e.g., pages, pictures, text, segments, etc.), where each object is characterized by a different "popularity".

An important technical issue in such broadcast systems is scheduling, i.e. the order in which data objects are transmitted. The server transmits objects, so that their mean aggregate reception delay is minimized for all users. This criterion, the minimized mean aggregate reception delay, is important not only for performance but for energy consumption as well: minimized delay implies that client devices are switched on for a minimized time, and thus, energy consumption of users is minimized.

In push-based broadcast systems, servers broadcast objects in periodic cycles consisting of T time units. Several algorithms have been used to construct the exact transmission schedule in a cycle, given as optimization criterion the minimization of the mean aggregate delay to start receiving an object, also called the access time (the results presented in the following, also apply to non-cyclic schedules, effectively when $T \to \infty$). The schedule of a cycle (period) includes multiple transmissions of each object, depending on its popularity [42]. It is known that, the optimal mean access time is achieved when all appearances of an object are equally distanced within the cycle [30].

Existing analyses of broadcast systems consider *memory-less* clients, i.e. clients which are not equipped with any storage. However, technological advances have led to the deployment of clients with memory today, and actually, memory sizes in the client devices are continuously increasing; in the following, we call this memory a *cache*, because it is not used for long-term storage. Existence of a cache at a client changes the model of a broadcast system, because a caching client is able to start collecting an object even if he/she turns on their device during the transmission of the desired object. This can be achieved provided that, each object transmission is done using packets that have headers with the appropriate packet information. This provision is the state of the art though, because often, transmission is done with fixed size cells, also called radio units, where each cell has an appropriate header, e.g. in GPRS [23], 802.11, etc.

In this chapter subsection, we present three main contributions. First, we provide a simple proof for the need of periodicity (equal distance in transmission) of popular objects in a cycle; the existent proof [30] uses arguments which are very complex. Second, in contrast to existing results, we consider the scheduling problem for caching clients. As we show, the existence of cache not only reduces the reception time of an object, but leads to an optimal broadcast schedule that is different from the optimal schedule for cache-less (traditional) clients. In our work, we calculate the reduced reception delay and we derive the i property that the optimal schedule for caching clients is based on, which is different from the one for cache-less clients. We also describe the scheduler that achieves the optimal schedule. For our work, we use as optimization parameter the mean aggregate reception delay, i.e. the sum of the access delay and the actual object reception time. In prior analyses [42],[46], where models similar to teletext were used, caching clients were not considered, because it was assumed that access time is significantly larger than actual object reception delay; however, in modern systems, this assumption does not hold because it is often necessary to transmit large objects with high popularity. Importantly, our optimization parameter reflects power consumption more realistically than existing models and calculations. This occurs because objects have variable sizes and thus, reception delay is not equal for all objects; however, prior work does not include this extra delay (and corresponding power consumption), because they consider environments with cache-less clients, where the only variable time is the access time.

Our third contribution refers to the analysis of pre-emptive scheduling, among other heuristics, for broadcast systems with or without caching clients. Since perfect periodicity of the broadcasting of all objects within a cycle is an NP-hard problem [9], we prove that the mean aggregate tuning delay of an object in a broadcast schedule may be further reduced, if we allow interruption (pre-emption) of an object's transmission in order to transmit on schedule another more popular one. This is contrary to the usual practice to transmit objects non-preemptively (without interruption). We deduce the conditions under which pre-emption is advantageous and we show that switching the transmission order of two consecutive objects is beneficial in some cases. Finally, we prove that interleaving transmission of two consecutive objects is not advantageous in any case; such interleaving is tempting, considering object packetization and the popularity of interleaving in ATM networks.

The paper is organized as follows. Subsection 3.2 presents an overview of data delivery architectures for broadcast systems and introduces the scheduling problem. Subsection 3.3 introduces the model of the broadcast system, which we analyze, and the notation used in the paper. Subsection 3.4 presents our simple proof of perfect periodicity. Subsection 3.5 presents our analysis for caching clients, including the calculation of aggregate reception delay and the scheduler for achieving optimal schedules in the new model. Finally, Subsection 3.6 describes the conditions under which pre-emptive scheduling provides improved results.

3.2 Data Delivery Architectures

Broadcast systems can be classified, in general, using 3 parameters:

- the data request model (push vs. pull);
- the timing properties of their scheduling scheme (periodic vs. aperiodic);
- the connection model between server(s) and client(s) (unicast vs. 1-to-N).

Every broadcast system is characterized by a specific choice for each of these parameters, leading to 8 possible system configurations. In the following, we elaborate on these three parameters.

A broadcast system is characterized by its data request model, where either the client requests (pulls) data from the server by posting a request, or the server sends (pushes) information to client(s) without explicit requests from the clients. In the pull model, the client posts an explicit request to the server, which, in turn, responds to the client with the requested data; i.e. the client initiates the data transfer. In contrast, in a push system, servers transmit data to clients without a prior request, using some schedule, i.e. the data transfer is initiated by the server itself. The push system is more appropriate for satellite broadcasting to mobile clients, because it eliminates the need for clients to transmit, which is power consuming, especially for the case of GEO satellites.

Data delivery can be periodic or aperiodic in a broadcast system. In periodic systems, data are transmitted according to a predefined schedule (off-line algorithm, similarly to the classic TDMA systems), while in aperiodic systems, data delivery is event-driven, i.e. a data transfer is performed when an event occurs (on-line algorithm, e.g., a request in a pull system or a server decision in a push system). Periodic systems with very long periods can be considered as aperiodic, although they are constructed using an off-line algorithm.

Furthermore, broadcast systems may use different connection models: data transfers may occur in a unicast fashion or in a multicast (broadcast) fashion. In unicast systems, data transfers between servers and clients occur over a one-to-one connection, where no other client can receive the transmitted data. In contrast, in 1-to-N systems, the delivered data are transferred to a set of N clients, which constitute a group. If N is the complete population, then the system follows a broadcast delivery method; however, the broadcast method is typically used in environments where N is unknown.

Several known systems can be classified using the above scheme. Pull-based systems are used, in general, for on-demand services, e.g. Video-on-Demand (VoD), news-on-demand, etc., where clients make requests for a specific service. Actually, these services are typically pull-based and aperiodic, since requests typically arrive at random time instances. Depending on the specific network configuration, the system can be based either on unicast or multicast (1-to-N) connections (a multicast connection is considered as broadcast, when N is the total population). Push-based systems include the characteristic examples of news services and newsgroup services as well as some forms of VoD, where clients have subscribed to receive specific information. Such systems can be either periodic

(sending specific information at designated time intervals) or aperiodic (sending update information only, or information at random intervals).

We focus on push-based, periodic, 1-to-N broadcast systems. These systems are popular in environments where N is unknown. We focus on periodic systems (off-line scheduling algorithms) but our results also apply to effectively non-periodic systems with very large periods. We focus on push-based, because we consider environments where data transmission from clients to servers is expensive and power hungry, e.g. satellites, and client-to-server communication occurs off-line, probably using a less expensive medium, such as a telephone connection. Scheduling in broadcast systems has received a lot of attention.

In pull-based systems, scheduling is necessary to specify the transmission sequence of objects (to choose the object to transmit next). Several scheduling algorithms have been developed for servers [5]: FCFS, Most Requested First (MRF), Most Requested First Lowest (MRFL), Longest Wait First (LWF) and RxW.

Several scheduling algorithms have been developed for push-based systems as well [4, 42, 41]. The criterion is to minimize average access time or tuning time.

One method to minimize access or tuning time, and thus power consumption, is to use indexing methods. In such environments, all objects are identified with a unique key and the system transmits indexing information along with the data; the index is a sequence of pairs of the form (key, location), where the key identifies an object and the location identifies the position of the object in a broadcast cycle. In this fashion, a client can tune in, find the appropriate index entry for the object required and then, it can tune in at the appropriate time, in order to receive the desired object. Several indexing methods have been developed, which differ according to the indexing method or the amount and transmission method of the indexing information. Methods such as (1,m)-Indexing, Distributed Indexing and Flexible Indexing [27] transmit index information in various ways (multiple transmissions of the complete index, distributed transmission of index portions, etc.), while hash-based techniques substitute the indexing information with hashing information that allows the client to calculate the appropriate position of an object in the broadcast cycle.

3.3 Model

We assume a **server** S that broadcasts information **objects** to a set of clients (users), who have subscribed for reception of specific objects off-line. We consider a Web-like environment, where the server transmits objects from a set $O = \{O_1, O_2, \ldots, O_N\}$ and each object O_i is characterized by a **popularity** p_i, which is a probability that quantifies the number of clients waiting to receive a specific object. Transmission is performed using fixed size cells, also called radio units. Objects have arbitrary length and l_i denotes the length of O_i, measured in radio units.

Data transmission is performed in periodic cycles T. In every cycle T, all objects are transmitted and each object may appear more than once in T, de-

pending on its popularity and its length. An appearance of an object in the broadcast cycle is denoted as an **instance** of the object. The **spacing** s_{ij} between two consecutive instances of an object O_i is the time between the beginning of the j-th instance and the beginning of the $(j+1)$-th instance.

Clients in the model are devices which are switched on arbitrarily in time. When a client is switched on, it remains on until it receives the desired object(s), and then it is turned off. The *switch-on* activity of a client can be modeled as posting a request for the object, which the client is waiting for; so, in the following, we will refer to the client switch-on as a *request*. We assume that, during a broadcast cycle T, client requests for an object O_i appear uniformly distributed.

We consider two different types of clients in the system: *caching* clients (equipped with cache) and *cache-less* ones. Cache-less clients need to receive an object from beginning to end, in order to consume it, because they do not have any storage capability. In contrast, caching clients have storage capability and can store partial object information; thus, they are able to store the last part of an object first and then wait for the reception of its beginning later.

We define as **tuning time** for an object, the **access time** of the object (the time until the beginning of reception of the object) plus the actual **reception time** of the object. Thus, we define *waiting time* differently from others, who consider the waiting time equal to the access time. This definition does not affect the construction of the optimal broadcasting schedule for cache-less clients, which are considered in prior work. In our work, we use as optimization parameter the *mean aggregate tuning time*, which is defined as the average of the mean tuning times of all users, i.e. $\sum p_i D_i$, where $i = 1, \ldots, M$, p_i is the *popularity* of object O_i (i.e., the fraction of the user population waiting for O_i at a given time) and D_i is the mean tuning time.

3.4 Periodic Object Transmission in a Cycle

Using the system model with cache-less clients, it has been shown that, for optimal broadcast scheduling (i.e., for minimization of the mean aggregate access delay), all instances of an object should be equally spaced within a cycle T [30].

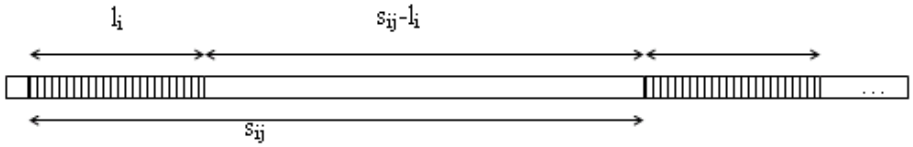


Fig. 22. Calculating the mean tuning delay of object O_i

In the following lemma, we provide a simple proof of this requirement, in contrast to [30], where the proof is quite obscure. Furthermore, we provide a proof

that the same requirement exists for optimal scheduling with caching clients who can start collecting radio units of the desired item as soon as they appear (they start storing parts of the desired object O_i, even if they are switched on during transmission of O_i).

Lemma 1. *The spacing s_i between any two consecutive instances of the same object O_i should be equal in a transmission cycle T.*

Proof. We consider two different cases, depending on whether clients have caches or not.

Cache-less clients

Assume that object O_i is been broadcasted f_i times in a broadcast cycle T. The instances of O_i are at spacings $s_{i1}, s_{i2}, \ldots, s_{if_i}$, where $\sum s_i = T$. If there is **one** request for O_i during T then there is a probability that it will appear during the spacing s_{i1}; this probability is s_{i1}/T, and the mean delay to receive all of O_i is $[l_i + (s_{i1}/2)]$. The same holds true for every spacing s_{ij}. Therefore, the average delay D_i to receive object O_i during a broadcast cycle T is:

$$\begin{aligned} D_i &= (1/T)\{s_{i1}[l_i + (s_{i1}/2)] + s_{i2}[l_i + (s_{i2}/2)] + \cdots + s_{if_i}[l_i + (s_{if_i}/2)]\} = \\ &= (1/T)\{s_{i1}^2/2 + s_{i2}^2/2 + \cdots + s_{if_i}^2/2 + l_i(s_{i1} + s_{i2} + \cdots + s_{if_i})\} = \\ &= (1/T)\{s_{i1}^2/2 + s_{i2}^2/2 + \cdots + s_{if_i}^2/2 + l_i T\} \end{aligned}$$

It is well known that the sum of squares of numbers with a constant sum is minimized when the numbers are equal. Therefore: $D_i = \min$ for $s_{i1} = s_{i2} = \cdots = s_{if_i} = T/f_i$, and so:

$$D_{i\min} = (f_i/2T)(T/f_i)^2 + l_i = (s_i/2) + l_i$$

Caching clients

Under the same assumptions as above (for cache-less clients), the average delay to receive object O_i in a broadcast cycle is:

$$\begin{aligned} D\text{cache}_i = (1/T)\Big\{ & l_i s_{i1} + (s_{i1} - l_i)[(s_{i1} - l_i)/2 + l_i] \\ & + l_i s_{i2} + (s_{i2} - l_i)[(s_{i2} - l_i)/2 + l_i] \\ & \cdots \\ & + l_i s_{if_i} + (s_{if_i} - l_i)[(s_{if_i} - l_i)/2 + l_i] \\ = (1/T)\Big\{ & (l_i \sum s_{ij}) + (s_{i1}^2/2) + (s_{i2}^2/2) + \cdots + (s_{if_i}^2/2) \\ & - l_i(\sum s_{ij}) + (l_i^2/2) f_i + l_i(\sum s_{ij}) - l_i^2 f_i \Big\} \\ = (1/T)\Big\{ & s_{i1}^2/2 + s_{i2}^2/2 + \cdots + s_{if_i}^2/2 + l_i(\sum s_{ij}) - (l_i^2/2) f_i \Big\} \\ = (1/T)\Big\{ & s_{i1}^2/2 + s_{i2}^2/2 + \cdots + s_{if_i}^2/2 + l_i T - (l_i^2/2) f_i \Big\} \end{aligned}$$

This expression is minimized when $s_{i1} = s_{i2} = \cdots = s_{if_i} = s_i = T/f_i$, and so:

$$\begin{aligned} D\text{cache}_{i\min} &= f_i (T/f_i)^2/2T + l_i - (l_i^2/2)(f_i/T) = (s_i/2) + l_i - (l_i^2 f_i/2T) \\ &= D_{i\min} - (l_i^2 f_i/2T) = D_{i\min} - (l_i^2/2s_i) \end{aligned}$$

The last expression shows that, local caching clients receive the desired object O_i with reduced (shorter) delay. The scheduling algorithm presented in [42], which requires that $s_i^2 p_i / l_i = \text{constant}$, does not take into account local memory (cache). If one wants to take advantage of a client's cache, one must modify the condition, on which the scheduling algorithm is based, accordingly. As we prove in the following subsection, the optimal broadcast schedule for caching clients requires that $s_i^2 p_i / l_i + l_i p_i = \text{constant}$. The difference between the two conditions becomes significant, if there exist lengthy objects with high popularities in the system.

3.5 Broadcast System with Caching Clients

Theorem 1. *The optimum broadcast schedule in a system with caching clients requires that*

$$s_i^2 p_i / l_i + l_i p_i = \text{constant}$$

Proof. Our proof follows the lines of the proof for cache-less clients [42] Assume that the following holds for all objects scheduled in a cycle T: the spacing s_i between all pairs of consecutive transmissions of the same object O_i within T is equal (as we will see this is not always possible, but this is approximately true for large broadcast cycles). Then, the average delay to receive object O_i is:

$$D\text{cache}_i = \big(s_i/2\big) + l_i - \big(l_i^2 f_i / 2T\big)$$

So, the mean aggregate delay for all items is:

$$\begin{aligned} D\text{cache} &= \sum D\text{cache}_i p_i = \\ &= \sum \big(s_i p_i/2\big) + \sum l_i p_i - \sum \big(l_i^2 f_i p_i / 2T\big) = \\ &= \sum l_i p_i + \big(1/2\big) \sum p_i \big[s_i - (l_i^2 f_i / T)\big] = \\ &= \sum l_i p_i + \big(1/2\big) \sum p_i l_i \big[(s_i/l_i) - (l_i f_i / t)\big] \end{aligned}$$

We denote as q_i the quantity $q_i = l_i f_i / T$. Clearly: $\sum q_i = \sum (l_i f_i / T) = T^{-1} \sum l_i f_i = 1$. Then, $s_i / l_i = s_i f_i / l_i f_i = T / l_i f_i = q_i^{-1}$. This results to:

$$D\text{cache} = \sum p_i l_i + (1/2) \sum p_i l_i [q_i^{-1} - q_i]$$

In order to find the q_i which minimize the mean aggregate delay Dcache, we set

$$(\partial D\text{cache} / \partial q_i = 0, \text{for } i = 1, 2, \ldots, M$$

So, we find that the following relation must be true:

$$p_1 l_1 (1 + q_1^{-2}) = p_2 l_2 (1 + q_2^{-2}) = \cdots = p_M l_M (1 + q_M^{-2}) = \text{constant}$$

This leads to:

$$\begin{aligned} p_i l_i (1 + q_i^{-2}) = p_i l_i + p_i l_i (T^2 / l_i^2 f_i^2) = p_i l_i + p_i l_i (s_i^2 f_i^2 / l_i^2 f_i^2) = \\ = p_i l_i + p_i s_i^2 / l_i = \text{constant} \end{aligned}$$

Considering the condition of the theorem, one can easily create an on-line scheduling algorithm that constructs optimal schedule for caching clients. We follow the method and the notation of [41]:
let Q denote the current time; the algorithm below decides which object to transmit at time Q. Let $R(j)$ denote the time at which an instance of object O_j was most recently transmitted and $H(j)$ denote the quantity $H(j) = \{[Q - R(j)]^2 p_j/l_j\} + p_j l_j, j = 1, 2, \ldots, M$ ($R(j) = -1$ if Q_j was never transmitted before).
On-line scheduling algorithm:
Step 1: Calculate $H_{\max} = \max\{H(j)\}$, for all j
Step 2: Choose O_i such that $H(i) = H_{\max}$ (if this equality holds true for more than one object, then choose one of them arbitrarily)
Step 3: Transmit object O_i at time Q
Step 4: $R(i) = Q$, go to Step 1

Observe that $[Q - R(j)]$ is the spacing between the current time and the time at which O_i was previously transmitted. The algorithm tries to keep constant the quantity

$$H(j) = \{[Q - R(j)]^2 p_j/l_j\} + p_j l_j$$

This quantity is similar to $p_i s_i^2/l_i + p_i l_i$, which must be constant according to the theorem.

3.6 Pre-emptive Object Scheduling

The optimum schedule requires that the consecutive instances of an object O_i are equally spaced within a broadcast cycle T. This is a very desirable property, especially for energy-limited users, because it reduces *busy-waiting* (as opposed to *stand-by* or *doze-waiting*) for the desired object. However, the design of an optimum schedule is an NP-hard problem [9], leading to use of heuristics that do not achieve the "perfect" schedule. So, as illustrated in Figure 23, it is very probable that an instance of an object (O_2 with length l_2 in the figure) will be broadcasted after the expected time (for perfect periodicity) with high probability, because the transmission of another object (O_1 in the figure) is in progress and must be completed first.

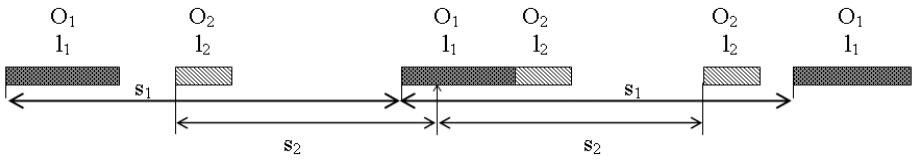


Fig. 23. A case where perfect scheduling is impossible

One can attempt to improve the schedule with the following heuristics:

1. Interrupt the transmission of object O_1, transmit part of O_2, complete the transmission of O_1 and then of O_2 (see Figure 24). As we prove below, this *interleaving* is not the right approach, because it always increases the total delay. It is easy to see that the same conclusion holds true if we attempt to do a finer interleaving.
2. Interrupt the transmission of O_1, transmit the complete O_2, and then resume and complete the transmission of O_1 (see Figure 25). This is an example of *pre-emptive transmission*. As we prove below, this approach may decrease the total delay under the right circumstances.
3. Transmit O_2 first (ahead of schedule) and then O_1 (behind schedule) (see Figure 3.6). Again, this ahead-of-schedule transmission decreases the total delay under certain conditions.

The results mentioned for each heuristic above hold for both client models, caching and cache-less. In the remaining subsection, we analyze all heuristics for the case of cache-less clients, because this analysis is much easier to present. We also present, for comparison, as case 4, the analysis of pre-emptive transmission for caching clients and derive the condition under which the total delay is reduced. In all cases analyzed below, only the delays of clients waiting for objects O_1 and O_2 are influenced. Remaining clients do not experience any difference in performance.

Case 1: Interleaving (Cache-less clients)

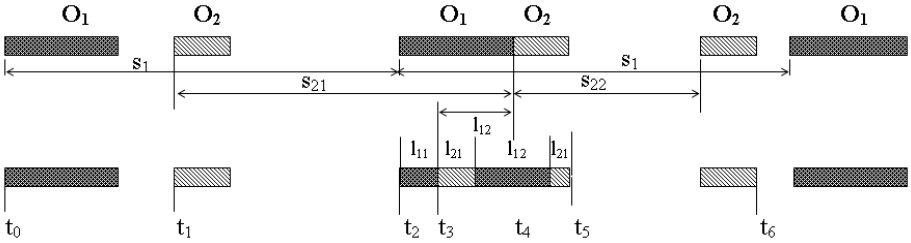


Fig. 24. Interleaving objects O_1 and O_2

Calculation of the increase of the average delay for object O_1:
Requests for object O_1 appearing in the interval $[t_0, t_2]$ will experience a delay **increase** to receive the object; this increase is equal to l_{21}. Remaining requests are unaffected. Therefore, the increase of the mean aggregate delay is equal to: $p_1 s_1 l_{21}/T$.

Calculation of the increase of the average delay for object O_2:
Requests for O_2 that appear in the interval $[t_3, t_4]$ will experience a delay **increase** to receive the object equal to: $t_6 - t_5 = s_{22}$ (s_{22} is the spacing between instances #2 and #3). The remaining requests are unaffected. Given that the interval $[t_3, t_4]$ has length l_{12}, the increase of the mean aggregate delay is equal to: $p_2 s_{22} l_{12}/T$.

Combining the results of the two previous calculations, we see that it is not appropriate to interrupt the transmission of O_1 in order to transmit part of O_2, because this decision always increases the mean aggregate delay by the amount:

$$\text{Total delay increase} = [p_1 s_1 l_{21} + p_2 s_{22} l_{12}]/T > 0$$

Case 2: Pre-emption (Cache-less Clients)
In this case, the transmission of O_1 is interrupted, O_2 is transmitted and then the transmission of O_1 is completed, as depicted in Figure 25.

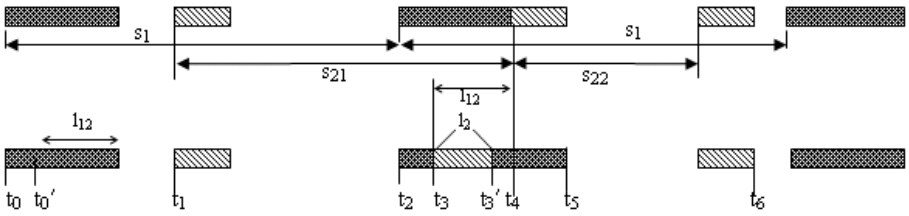


Fig. 25. Pre-emptive transmission of O_2

Calculation of the increase of the average delay for object O_1:
Requests for O_1 appearing in the interval $[t_0, t_2]$ will experience a delay **increase** equal to l_2, in order to receive O_1. The remaining requests are unaffected. Therefore, the increase of the mean aggregate delay for O_1 is equal to: $p_1 s_1 l_2/T$.

Calculation of the increase of the average delay for object O_2:
Requests for O_2 appearing in the interval $[t_1, t_3]$ (with length $s_{21} - l_{12}$) will experience a delay **decrease** equal to l_{12}, in order to receive O_2. Requests made during $[t_3, t_4]$ will experience a delay **increase** equal to s_{22} to receive O_2. The remaining requests are unaffected. Therefore, the **increase** of the mean aggregate delay is:

$$[-p_2(s_{21} - l_{12})l_{12} + p_2 l_{12} s_{22}]/T$$

Given the above calculations, the total **increase** of the mean aggregate delay ΔD is:

$$\begin{aligned} \Delta D &= [p_1 s_1 l_2 - p_2(s_{21} - l_{12})l_{12} + p_2 l_{12} s_{22}]/T = \\ &= [p_1 s_1 l_2 + p_2 l_{12}^2 - p_2(s_{21} - s_{22})l_{12}]/T \end{aligned}$$

So, the delay increase depends on l_{12}. If we set $d\Delta D/dl_{12} = 0$, then we find that ΔD takes a minimum value, if $l_{12} = (s_{21} - s_{22})/2$. This minimum value is:

$$\Delta D_{\min} = [p_1 s_1 l_2 - p_2 (s_{21} - s_{22})^2]/4T$$

and is negative (= **maximum decrease of delay**) if

$$(s_{21} - s_{22})^2 > 4 p_1 s_1 l_2 / p_2$$

Thus, if this inequality holds, then we can reduce the mean aggregate delay, if we interrupt the transmission of O_1 after $l_1 - l_{12} = l_1 - (s_{21} - s_{22})/2$ radio units, in order to transmit O_2.

Case 3: Ahead-of-Schedule Transmission (Cache-less Clients)
In this case, we transmit O_2 ahead of schedule and O_1 behind schedule.

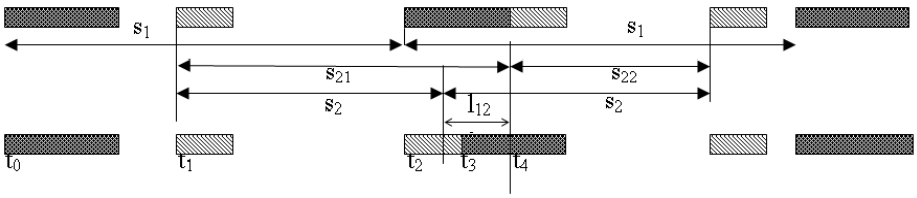


Fig. 26. Switching the transmission order of O_1 and O_2

Calculation of the increase of the average delay for object O_1:
Requests for O_1 appearing in the interval $[t_0, t_2]$ will experience a delay **increase** equal to l_2, in order to receive O_1. Requests in the interval $[t_2, t_3]$ will experience a delay **decrease** equal to $(s_1 - l_2)$, while all remaining requests are unaffected. Therefore, the increase of the mean aggregate delay is equal to:

$$[p_1 s_1 l_2 - p_1 l_2 (s_1 - l_2)]/T$$

Calculation of the increase of the average delay for object O_2:
Requests for O_2 appearing in the interval $[t_1, t_2]$ will experience a delay **decrease** equal to l_1 to receive O_2. Requests in the interval $[t_2, t_4]$ will experience a delay **increase** equal to s_{22}. All remaining requests are unaffected. Therefore, the increase of the mean aggregate delay is equal to:

$$[-p_2 l_1 (s_{21} - l_1) + p_2 l_1 s_{22}]/T$$

The total delay increase is:

$$\begin{aligned} \Delta D &= [p_1 s_1 l_2 - p_1 l_2 (s_1 - l_2) - p_2 l_1 (s_{21} - l_1) + p_2 l_1 s_{22}]/T = \\ &= [p_1 l_2^2 + p_2 l_1^2 - p_2 l_1 (s_{21} - s_{22})]/T \end{aligned}$$

This expression becomes negative (**decrease of the delay**) if

$$(s_{21} - s_{22}) > l_1 + (p_1 l_2^2)/p_2 l_1$$

Since $(s_{21} - s_{22}) = 2l_{12}$ (see Figure 3.6), we obtain:

$$l_{12} > [l_1 + (p_1 l_2^2)/p_2 l_1]/2$$

If this inequality holds, the mean aggregate delay is **reduced** when the transmission order of O_1 and O_2 is switched.

The results of this analysis hold true for the case of caching clients as well; however, the analysis is more involved. We present the analysis of pre-emptive transmission with caching clients.

Case 4: Pre-emption (Caching Clients)
Consider the pre-emptive transmission of object O_2, shown in Figure 25, for the case of caching clients. We derive the conditions under which this pre-emptive transmission reduces the total tuning time.

Calculation of the increase of the average delay for object O_1:
Clients requesting O_1 and appearing in the interval $[t'_0, t_2]$ will experience an **increase** of the tuning time by l_2. Requests during the interval $[t_2, t_3]$ will not be affected. Requests during $[t_3, t'_3]$ will have a mean **decrease** of their tuning time by $l_2/2$. Similarly, requests during $[t'_3, t_4]$ will have a **decrease** of their tuning time by l_2, while requests during $[t_4, t_5]$ will have a mean decrease by $l_2/2$. Putting these together, we see that the mean increase of the tuning time of clients requesting O_1, due to the pre-emptive transmission of O_2 is:

$$\Delta D_1 = (p_1/T) l_2[(s_1 - l_1 + l_{12}) - l_2/2 - (l_{12} - l_2) - l_2/2] = p_1 l_2 (s_1 - l_1)/T$$

Calculation of the increase of the average delay for object O_2:
Clients requesting O_2 and arriving in the interval $[t_1, t_3]$ will have a **decrease** of their tuning time by l_{12}. Requests during the interval $[t_3, t'_3]$ will have a mean **increase** of their tuning time by $(s_{22} - l_2/2)$, requests during the interval $[t'_3, t_4]$ an **increase** by s_{22}, and requests during the interval $[t_4, t_5]$ a mean **increase** by $l_2/2$. Thus, the mean **increase** of tuning time for object O_2 is:

$$\begin{aligned}\Delta D_2 &= (p_2/T)[-(s_{21} - l_{12}) l_{12} + (s_{22} - l_2/2) l_2 + (l_{12} - l_2) s_{22} + l_{22}/2] = \\ &= p_2 l_{12} (s_{22} - s_{21} + l_{12})/T\end{aligned}$$

Working similarly to the case of cache-less clients, we find that the condition under which pre-emptive transmission **reduces** the total tuning time is:

$$(s_{21} - s_{22})^2 > 4 p_1 (s_1 - l_1) l_2 / p_2$$

Thus, the maximum reduction of the tuning time is achieved when: $l_{12} = (s_{21} - s22)/2$.

3.7 Conclusions

We showed that, as expected, a local cache reduces the time required for the reception of an object. We also proved that the optimal broadcast schedule for these caching clients is different from the optimal schedule for cache-less clients (it must satisfy the condition $s_i^2 p_i/l_i + p_i l_i = \text{constant}$, rather than the condition $s_i^2 p_i/l_i = \text{constant}$). Considering a broadcast schedule constructed with heuristic algorithms that are based on these conditions, we proved that, for caching or cache-less clients, the mean aggregate delay for the complete reception of an object may be further reduced, if we allow the interruption of the transmission of an object in order to transmit on schedule another more popular one. We deduced the conditions under which this pre-emption reduces the mean aggregate delay. Furthermore, we showed that under some conditions the switching in the transmission order of two neighboring objects may also be advantageous. Finally, we proved that the interleaving of the transmissions of two neighboring objects is not beneficial in any case.

4 Acknowledgments

The authors would like to thank professor Jon Crowcroft for his detailed comments.

References

[1] Freeflow: How it works. Akamai, Cambridge, MA, USA, November 1999.

[2] C. C. Aggarwal, J. L. Wolf, and P. S. Yu. A permutation-based pyramid broadcasting scheme for video-on-demand systems. In *proceedings of ICMCS*, pages 118–126, June 1996.

[3] Charu C. Aggarwal, Joel L. Wolf, and Philip S. Yu. On optimal batching policies for video-on-demand storage servers. In *Proceedings of ICMCS*, pages 253–258, Hiroshima, Japan, June 1996.

[4] D. Aksoy, M. Altinel, R. Bose, U. Cetintemel, M.J. Franklin, J. Wang, and S.B. Zdonik. Research in Data Broadcast and Dissemination. In *Proceedings of the First International Conference on Advanced Multimedia Content Processing (AMCP'98)*, Lecture Notes in Computer Science, pages 194–207. Springer Verlag, 1999.

[5] D. Aksoy and M. Franklin. Scheduling for Large-Scale On-Demand Data Broadcasting. In *Proceedings of INFOCOM 1998*, San Francisco, CA, 1998.

[6] K. C. Almeroth and M. H. Ammar. On the use of multicast delivery to provide a scalable and interactive video-on-demand service. *IEEE Journal on Selected Areas in Communications*, 14(6):1110–1122, April 1996.

[7] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proceedings of ACM SIGCOMM*, August 2002.

[8] Sujata Banerjee, Jack Brassil, Amy C. Dalal, Sung Ju Lee, Ed Perry, Puneet Sharma, and Andrew Thomas. Rich media from the masses. Technical Report HPL-2002-63R1, HP Lab, May 2002.

[9] A. Bar-Noy, B. Randeep, J. Naor, and B. Schieber. Minimizing service and operation cost of periodic scheduling. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 11–20, 1998.

[10] Ernst Biersack, Alain Jean-Marie, and Philippe Nain. Open-loop video distribution with support of vcr functionnality. *Performance Evaluation*, 49(1-4):411–428, September 2002.

[11] Yitzhak Birk and Ron Mondri. Tailored transmissions for efficient near-video-on-demand service. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, pages 226–231, june 1999.

[12] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron, and Atul Singh. Splitstream: High-bandwidth content distribution in a cooperative environment. In *proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS)*, March 2003.

[13] G. Chan and F. Tobagi. Distributed servers architectures for networks video services. *IEEE/ACM Transactions on Networking*, 9(2):125–136, April 2001.

[14] Yang H. Chu, Sanjay G. Rao, Srinivasan Seshan, and Hui Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *proceedings of ACM SIGCOMM*, San Diago, CA, August 2001.

[15] Dash Optimization. *Xpress-Mp Essentials*, 2001.

[16] DirecPC. URL: `http://www.direcpc.com/index.html`.

[17] D. Eager, M. Vernon, and J. Zahorjan. Optimal and efficient merging schedules for video-on-demand servers. In *proceedings of the 7th ACM Multimedia Conference*, November 1999.

[18] Derek Eager, Mary Vernon, and John Zahorjan. Minimizing bandwidth requirements for On-Demand data delivery. *IEEE Transactions on Knowledge and Data Engineering*, 2001.

[19] L. Gao and D. Towsley. Threshold-based multicast for continuous media delivery. *IEEE Transactions on Multimedia*, 3(4):405–414, December 2001.

[20] V.K. Goyal. Multiple description coding: compression meets the network. *IEEE Signal Processing Magazine*, 18(5):74 –93, September 2001.

[21] Yan Guo, Subhabrata Sen, and Don Towsley. Prefix caching assisted periodic broadcast: Framework and techniques for streaming popular videos. In *proceedings of IEEE ICC*, April 2002.

[22] Yang Guo, Kyoungwon Suh, Jim Kurose, and Don Towsley. P2cast: Peer-to-peer patching scheme for vod service. In *Proceedings of the 12th World Wide Web Conference (WWW)*, Budapest, Hungary, May 2003.

[23] G. Heine. *GPRS from A to Z*. Artech House Inc., April 2000.

[24] Ailan Hu. Video-on-Demand broadcasting protocols: A comprehensive study. In *proceedings of Infocom*, volume 1, pages 508–517, Anchorage, Alaska, USA, April 2001.

[25] K. A. Hua and S. Sheu. Skyscraper broadcasting for metropolitan vod. In *proceedings of Sigcomm*, August 1997.

[26] Kien A. Hua, Ying Cai, and Simon Sheu. Patching : A multicast technique for true video-on-demand services. In *proceedings of ACM Multimedia*, pages 191–200, 1998.

[27] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Energy Efficient Indexing on Air. *ACM SIGMOD Record*, 23(2):25–36, June 1994.

[28] Infogate. URL: `http://www.pointcast.com/`.

[29] Intracom S.A. URL: `http://www.intranet.gr/en/products/internet/ios.htm`.

[30] R. Jain and J. Werth. Airdisks and airRAID: Modeling and scheduling periodic wireless data broadcast. Technical report, DIMACS Technical Report 95-11, May 1995.

[31] John Jannotti, David K. Gifford, and Kirk L. Johnson. Overcast: Reliable multicasting with an overlay network. In *proceedings of the 4-th Symp. on Operating Systems Design and Implementation*. Usenix, Octobre 2000.

[32] L. Juhn and L. Tseng. Fast data broadcasting and receiving scheme for popular video service. *IEEE Trans.on Broadcasting*, 44(1):100–105, March 1998.

[33] Li-Shen Juhn and Li-Ming Tseng. Harmonic broadcasting for video-on-demand service. *IEEE Transactions on Broadcasting*, 43(3), September 1997.

[34] K. Kong and D. Ghosal. Mitigating server-side congestion in the internet through pseudoserving. *IEEE/ACM Transactions on Networking*, pages 530–544, August 1999.

[35] J. Nussbaumer, B. Patel, F. Schaffa, and J. Sterbenz. Networking requirements for interactive video on demand. *IEEE Journal on Selected Areas in Communications*, 13(5):779–787, 1995.

[36] V. N. Padmanabhan, Helen J. Wang, Philip A. Chou, and Kunwadee Sripanidkulchai. Distributing streaming media content using cooperative networking. In *proceedings of NOSSDAV*, May 2002.

[37] Sridhar Ramesh, Injong Rhee, and Katherine Guo. Multicast with cache (mcache): An adaptive zero delay video-on-demand service. *IEEE Transactions of Circuit and System of Video Transmission*, 11(3), March 2001.

[38] S. Shekhar and D. Liu. Genesis and Advanced Traveler Information Systems ATIS: Killer Applications for Mobile Computing. MOBIDATA Workshop, 1994.

[39] Simon Sheu and Kien A. Hua. Virtual batching: A new scheduling technique for video-on-demand servers. In *Database Systems for Advanced Applications*, pages 481–490, April 1997.

[40] TiVo. What is tivo: Technical aspects, 2003.

[41] N.H. Vaidya and S. Hameed. Data broadcast in asymmetric wireless environments. In *Proceedings of Workshop on Satellite-based Information Services (WOSBIS)*, New York, November 1996.

[42] N.H. Vaidya and S. Hameed. Scheduling data broadcast in asymmetric communication environments. *ACM/Baltzer Wireless Networks*, 5(3):171–182, 1999.

[43] S. Viswanathan and T. Imielinski. Pyramid broadcasting for video on demand service. In *Proceedings of Multimedia Conference*, San Jose, CA, February 1995.

[44] Bing Wang, Subhabrata Sen, Micah Adler, and Don Towsley. Proxy-based distribution of streaming video over unicast/multicast conncetions. In *proceedings of Infocom*, june 2002.

[45] Paul P. White and Jon Crowcroft. Optimized batch patching with classes of service. *ACM Communications Review*, October 2000.

[46] J. W. Wong. Broadcast delivery. *Proceedings of the IEEE*, 76:1566–1577, December 1999.

[47] Dongyan Xu, Heung-Keung Chai, Catherine Rosenberg, and Sunil Kulkarni. Analysis of a hybrid architecture for cost-effective streaming media distribution. In *proceedings of SPIE/ACM Conference on Multimedia Computing and Networking (MMCN)*, Santa Clara, CA, January 2003.

[48] Y. Zhao, D. Eager, and M. Vernon. Network bandwidth requirements for scalable on-demand streaming. In *proceedings of Infocom*, june 2002.

Algorithms for Scalable Content Distribution

Ernst W. Biersack (Ed.)[1], Anwar Al Hamra[1], Guillaume Urvoy-Keller[1], David Choi[1], Dimitrios N. Serpanos[2], and Apostolos Traganitis[3]

[1] Institut Eurecom, Departement of Corporate Communications
B.P. 193, 06904 Sophia Antipolis, France
{erbi, alhamra, keller, choi}@eurecom.fr
[2] University of Patras
Department of Electrical and Computer Engineering
serpanos@ee.upatras.gr
[3] University of Crete, Department of Computer Science
tragani@csd.uoc.gr

Abstract. In this chapter, we address how to achieve scalable content distributions. We present two contributions, each of which uses a different approach to distribute the content.
In the first part of this chapter, we consider a terrestrial overlay network and build on top of it a VoD service for fixed clients. The goal is to minimize the operational cost of the service. Our contibutions are as follows. First, we introduce a new video distribution architecture that combines open-loop and closed-loop schemes. This combination makes the overall system highly scalable, very cost-effective, and ensures a zero start-up delay. Our second contribution is a mathematical model for the cost of delivering a video as a function of the popularity of that video. Our analytical model, along with some extensions, allows us to explore several scenarios: (i) long videos of 90 min (movies), (ii) short videos of a few min (clips), (iii) the dimensioning of a video on demand service from scratch, and (iv) the case of the optimization of an already installed video on demand service (i.e. the limited resources scenario).
In the second part of this chapter, we consider a satellite distribution of contents to mobile users, or in general to users thar are occasionally connected. We consider a push-based model, where the server periodically downloads objects. We assume that clients register to the service off-line. Our goal is to minimize the mean aggregate reception delay over all objects where each object is weighted by its popularity. Our contibutions in this part are as follows. First, we provide a simple proof for the need of periodicity (equal distance in transmission) of popular objects in a cycle. Second, in contrast to existing results, we consider the scheduling problem for caching clients. To increase the performance of the system, we also evaluate a pre-emptive scheduling algorithm that allows interruption (pre-emption) of an object's transmission in order to transmit on schedule another more popular one.

[1] Institut Eurécom's research is partially supported by its industrial members: Bouygues Télécom, Fondation d'entreprise Groupe Cegetel, Fondation Hasler, France Télécom , Hitachi, ST Microelectronics, Swisscom, Texas Instruments, Thales

M. Smirnov et al. (Eds.): COST 263 Final Report, LNCS 2856, pp. 207–262, 2003.

1 Introduction

The Internet has evolved, in the last decade, from a research oriented network to a large scale commercial network. Still, most of the trading concerns non-real time documents or services and only two parties are involved in general (dealer/client). The deployment of content distribution networks faces two main obstacles:

- There currently exists no scalable and widely deployed means to serve simultaneously a great number of clients;
- The wide heterogeneity of clients, not only in terms of access bandwidth, but also in terms of reachability since clients might be fixed or mobile.

To achieve a scalable distribution (whose cost grows less than linearly with the number of clients), two approaches have been commonly used: (i) a dedicated overlay network to provide a multicast service at the application layer or (ii) a satellite distribution whose cost is essentially independent from the number of simultaneous clients. In this chapter, we present two contributions that use either one of these scalable distribution techniques with different hypotheses concerning the clients:

In the first part of this chapter, we use a terrestrial overlay network and build on top of it a video on demand service for fixed clients. Our objective is to minimize the operational cost of the service. To minimize the distribution (bandwidth) cost, we use an open-loop central server since its cost is roughly independent of the number of simultaneously connected clients. The only limitation of open-loop algorithms is that they induce a non-zero start-up delay, which accounts for the duration between the moment where the client starts receiving the video stream and the moment it is able to visualize the video on the screen. To achieve a zero start-up delay, we thus introduce servers that stream the beginning of the video (which is referred to as prefix) immediately upon connection of the client to the video on demand service. To build a cost-optimal video on demand service, we thus end up solving an optimization problem whose variables are: the popularity of the requested video, the number of video servers that we use to service the beginning of the video and the fraction of the video that the "prefix" represents.

Our contribution is the design of a mathematical model that allows to find the architecture (i.e. the optimal values of these variables) that minimizes the total operational cost of our video on demand service. In contrast to previous studies, our model considers realistic underlying physical network infrastructures to accurately model all bandwidth costs and also includes both the I/O and storage requirements. Our analytical model, along with some extensions, allows us to explore several scenarios: (i) long videos of 90 min (movies), (ii) short videos of a few min (clips), (iii) the dimensioning of a video on demand service from scratch, and (iv) the case of the optimization of an already installed video on demand service (i.e. the limited resources scenario).

In the second part of this chapter, we consider a completely different setting since we assume a satellite distribution of content to mobile users that are occasionally connected. We consider a push model where a client has registered off-line to a service that can be modeled as the periodic download of objects representing, for instance, pages, pictures, or short videos. The objective is to minimize the mean aggregate reception delay of all objects where an object is weighted by its popularity. This metric is important since it ensures minimization of the energy consumption at the client side. Existing studies on broadcast systems generally assume that a client is not able to temporarily cache an object. In contrast to these studies we consider the case where clients are equipped with a cache, which is a realistic assumption with respect to the recent evolution of technology.

Our contributions in the second part of this chapter are the following. We first derive a simpler proof than the existing one justifying the periodic transmission of objects during a cycle is optimal for the case of cache-less clients. We then extend this proof to the case of caching clients. We also devise the corresponding optimal algorithm that computes the optimal schedule of each object. In practice, however, achieving a perfect periodicity of the broadcasting of objects within a cycle is an NP-hard problem because the optimal time to transmit an object may lie before the end of transmission of the previous one.

To improve the performance of the system, we investigate several empirical pre-emption strategies, some of them resulting in a real performance improvement while others, like the ones based on interleaving transmissions of multiple objects are proved to be inefficient.

2 Cost-Optimal Dimensioning of a Large Scale Video on Demand System

2.1 Introduction

Background Video streams such as MPEG-2 encoded video require several Mbps and providing a VoD service to a large number of clients poses high resource demands to the server and the network. The bandwidth-intensive nature of video requires efficient distribution techniques that typically serve multiple clients who request the same video at approximately the same time via a single video stream that is multicast. VoD systems can be classified in *open–loop systems* [6, 43, 2, 25, 32, 33, 11] and *closed–loop systems* [3, 39, 26, 19, 37, 17].

- Open-loop VoD systems partition each video into smaller pieces called *segments* and transmit each segment at its assigned transmission rate. The first segment is transmitted more frequently than later segments because it is needed first in the playback. All segments are transmitted periodically and indefinitely. In open–loop systems there is no feedback from the client to the server and transmission is completely one–way. Open-loop systems are suitable for popular videos where multicast is efficient. However, open-loop

systems introduce a start-up delay[4] and waste bandwidth in the case of non-popular videos.
- Closed-loop systems, on the other hand, require the client to contact the server. Closed–loop systems generally open a new unicast/multicast stream each time a client or a group of clients issue a request for a video. Whenever a new client arrives, the client joins an ongoing multicast stream that has been initiated for earlier clients, if there is any, and retrieves the missed part due to its later arrival via unicast, which ensures an immediate playout of the video. More often, Closed-loop schemes are suitable for videos with moderate popularity where the load on the server is reasonably low.

In the first part of this chapter, we present a new video distribution architecture that combines both, open-loop and closed-loop systems. This combination makes our video distribution architecture suitable for both, popular and non-popular videos. Moreover, having used a closed-loop system to deliver the first part of the video, our video distribution architecture ensures a zero start-up delay.

Contributions and Related Work Providing an efficient and scalable video distribution to a large client population has been investigated extensively over the years. The basic idea to achieve scalability is to serve multiple clients via a single stream, using multicast. Open-loop schemes take a full advantage of multicast. *Staggered broadcasting* [6] is the straightforward open-loop scheme where the server allocates for each video C channels each of bandwidth equal to the play back rate b of the video. On each channel, the whole video is broadcast at rate b. The starting points of the transmission on the different channels are shifted to guarantee a start-up delay of no more than L/C, where L is the length of the video. Clients listen to only one channel at the same time and no storage capacity is needed at the client side. The start-up delay can be improved only by increasing the number of allocated channels C.

More efficient and complicated schemes have been proposed later on. *Pyramid Broadcasting* (PB) [43] divides the video into C segments of increasing size, where C is also the total number of logical channels. The size of each segment is made α times larger than the size of the previous segment ($L_i = \alpha L_{i-1},\ i > 1$). The value of α is set to $\frac{B}{K \cdot C}$, where K is the total number of videos in the system and B is the total bandwidth over all the C channels. Note that B is divided equally amongst the C channels. Segments i for all videos are multiplexed into the same channel i and broadcast consecutively and periodically on that channel i at rate B/C. At any moment, the client downloads from at most two channels. This scheme reduces the bandwidth utilization and the start-up delay as compared to the *staggered broadcasting* [6] while guaranteeing a continuous playback of the video. However, its main disadvantage is that segments are transmitted on the different channels at a high rate (i.e. B/C) which requires a high I/O capacity at the client side.

[4] We ignore here the *transmission* delay due to sending a request to a server or joining a multicast group.

Permutation-based Pyramid Broadcasting (PPB) [2] addresses the problem of high I/O requirement at the expense of a larger start-up delay and a more complex synchronization. PPB uses the same geometric series proposed by PB to define the length of each segment ($L_i = \alpha L_{i-1},\ i > 1$). Unlike PB, PPB proposes to divide each channel into $K \cdot p$ sub-channels, where p sub-channels are dedicated for the same video segment. Then, each segment is broadcast on p sub-channels in such a way that the access time of segment i is L_i/p. Thus, segments are transmitted at rate $B/(C \cdot K \cdot p)$ instead B/C in the case of the PB scheme. On the other hand, at any moment, clients download from at most two separate sub-channels.

Another efficient scheme is the *skyscraper Broadcasting* (SB) [25]. SB uses a recursive function instead of a geometric series to generate the segment lengths. Each segment is then broadcast on a separate channel at the playback rate b. Clients listen to at most two channels at the same time. This scheme accounts for buffer space limitations of the clients by constraining the size of the last segment.

Fast Broadcasting (FB) [32] divides each video into segments according to a geometric series ($L_i = 2^i$). Each segment is broadcast on a separate channel at the playback rate b. Clients listen to all channels at the same time. FB provides the lowest bandwidth requirement at the server as compared to the above schemes.

Harmonic Broadcasting (HB) [33] presents another variant from the same general idea. In this scheme, the video is divided into N segments of equal size. Segment i is transmitted at the rate $1/i$ and clients download from all channels at the same time.

All the schemes cited above are constrained to highly regular designs which limits their flexibility. The *Tailored Transmission Scheme* [11] is a more flexible organization that can be adapted to meet different constraints in the system such as limited I/O capacities of the server/clients, limited storage capacity at the client side. This scheme will be detailed more in subsubsection 2. For more details on open-loop schemes, see [24].

While open-loop schemes broadcast the video regardless the request pattern of the clients, closed-loop schemes serve the video in response to client requests. Closed-loop schemes can be classified into batching, patching, and hierarchical merging schemes. The basic idea of batching [3, 39] is that the server groups clients that arrive within a given interval of time, in order to serve them via a single multicast stream. However batching introduces a start-up delay.

Patching techniques [26], on the other hand, ensure a zero start-up delay. The first client that requests a video receives a complete stream for the whole video from the server. When a new client arrives after the first one, we distinguish two cases:

– The complete stream that has been initiated for the first client is still active. In this case, the client needs to connect to that stream which changes from unicast to multicast. In addition, the client receives immediately from the

server a unicast patch for the part it missed in the complete stream due to its late arrival.
- There is no active complete stream. In this case, the server initiates a new one and the process repeats for clients that arrive later.

Note that clients that are listening to the same complete stream form a session. One efficient extension of patching has been proposed in [19] with the introduction of a threshold policy to reduce the cost of the unicast patches. Whenever a new client arrives and a complete stream is active, the threshold value serves to decide whether to initiate a new complete stream for that client, or whether that client must join the last ongoing complete stream. This scheme will be explained more in subsubsection 2.

White et al. [45] propose OBP, a hybrid scheme that combines patching and batching techniques. As for the classical patching [26], in OBP, the server initiates a new complete stream for the first client. A later client either receives a new complete stream in case there is no active complete stream, or the client joins an already existing one. In this later case, in contrast to patching, the client does not receive immediately the missed part in the complete stream; instead, the server batches many clients that arrive within a given interval of time and serve them via multicast. Thereby, as compared to patching, OBP reduces the delivery cost of the missed parts at the expense of a larger commencement viewing delay of the video. Similar to the controlled multicast, the authors introduce a threshold to decide when a new session should start. They also extend the OBP to deal with the case of heterogeneous clients where each client chooses the start-up latency according to the price it is willing to pay for the service.

Hierarchical merging [17] is a more efficient closed-loop scheme. As its name indicates, this scheme merges clients in a hierarchical manner. When a new client arrives, the server initiates a unicast stream to that client. At the same time, the client listens to the closest stream (target) that is still active. When the client receives via unicast all what it missed in the target stream, the unicast stream is terminated and the client merges into the target stream, and the process repeats. It has been shown that the server bandwidth for the hierarchical merging increases logarithmically with the number of clients.

In this work, we propose a scalable and efficient video distribution architecture that combines open-loop and closed-loop mechanisms to assure a zero start-up delay. Each video is partitioned into a prefix and a suffix. The suffix is stored at a central server, while the prefix is stored at one or more prefix servers. A client who wants to view a video joins an already on-going open-loop multicast distribution of the suffix while immediately requesting the prefix of the video as a patch [26] that is sent either via unicast or multicast [19]. We develop an analytical model for that video distribution architecture that allows to compute for a video with a given popularity the cost-optimal partitioning into prefix and suffix and the placement of the prefix servers in the distribution tree.

In contrast to previous studies (see for example [13, 21, 44]), we

- Model the network as a tree with outdegree m and l levels. In comparison, Guo et al. [21] consider only a two-level distribution architecture.

- Account in the model of the network transmission cost for the number of clients that are simultaneously served by the multicast distribution (either from the prefix servers or the suffix server).
- Allow for the prefix servers to be placed at any level in the distribution tree and not only at the last hop between client and network [44].
- Include in our cost model not only the network transmission cost but also the *server* cost, which depends on both, the storage occupied and the number of input/output streams needed. While the network transmission cost is a major cost factor, the server cost must be included in the overall cost model, especially when we try to design a cost-optimal video distribution architecture. Otherwise, independent of the popularity of a video, the obvious/trivial architecture will be the one where a large number of prefix servers are placed near the clients. While previous papers [21] have treated in their model the storage space of the prefix servers as a scarce resource, we feel that the cost model can be made more realistic by explicitly modeling the cost of the prefix servers.

A related cost model has been presented previously in [35]. The authors model the distribution network as a tree with l levels. In this model, *caches* can be placed at any level in the network. However, these caches can only store the entire video. Our model is more flexible in the sense that any portion of the video can be stored at the prefix servers. The system cost is simply the sum over the storage cost, the I/O bandwidth cost of server/caches, and the network bandwidth cost. Moreover, the cost formulas developed are for simple scenarios with only unicast server/clients connections.

A recent paper by Zhao et al. [48] looks at different network topologies, such as fan-out K, daisy-chain, balanced tree and network topologies generated with topology generators. The analysis focuses on schemes that provide an instantaneous delivery of the video. They derive a tight bound on the minimum network bandwidth requirement for each topology. They show that, at best, the minimum network bandwidth requirement scales as $\mathrm{O}(lnN)$, where N is the average number of clients during an interval of time equal to the duration of the video. They also show that it is possible to achieve simultaneously close to the minimum network and server bandwidth usage with practical distribution protocols.

The rest of this sub-chapter is organized as follows: in Subsection 2.2, we present the broadcasting algorithm and content distribution overlay that we use to build the video distribution service. In Subsection 2.3, we present the basic PS-model that allows to obtain the cost formulas for the single video case when there is no a priori constraint on the content distribution network. In Subsection 2.4, we apply the PS-model to the case of long videos, and investigate different scenarios such as heterogeneous or homogeneous bandwidth costs, zero servers costs, etc.. In Subsection 2.5, we prove that the PS-model is still advantageous in the case of short videos, e.g. news clips. In Subsection 2.6, we first study the dimensioning of the system with n videos and no resource constraints. We next devise the algorithm that optimally assigns video to prefix servers in the

case of limited resources (I/O, storage) and fixed placement of these servers. We conclude this sub-chapter in Subsection 2.7.

2.2 The System Environment

Prefix Caching Assisted Periodic Broadcast Prefix caching assisted periodic broadcast[5] [21] assumes that clients are serviced by a main central suffix server and also by local prefix servers, which can be located throughout the network. A video is partitioned into two parts, the prefix and the suffix, which can be of arbitrary proportion. We denote by L (min) the length of the video and D (min) the length of the prefix. Hence, the suffix will have the length $L - D$. The entirety of the prefix is always *viewed before* the suffix. The main idea of the broadcast scheme is that prefix and suffix transmissions should be decoupled in order to transmit each most effectively. The reason why the prefix and suffix are transmitted differently is that the client must receive the prefix *immediately upon request* while the suffix needs not to be received until the prefix has been completely viewed.

Because the prefix must be immediately received, there is less flexibility in the choice of a transmission scheme for the prefix. In addition, transmitting the prefix from the central server to each client may be costly. In order to reduce transmission costs, the prefix can be stored locally at multiple prefix servers, which can more cheaply transmit the prefix to their local audiences. For the suffix, on the other hand, there is more leeway in the method of broadcast since it needs not to be received immediately. The allowable delay D in transmitting the suffix permits to deliver it with an open-loop scheme. Therefore, the suffix is retained at the central server, benefiting from sharing amongst a relatively larger number of clients as well as avoiding the server costs incurred to replicate data across multiple servers.

Once specific transmission schemes for prefix and suffix have been chosen, the remaining design parameters are the length of the prefix (and suffix) and the height of placement of the prefix servers in the network. The prefix length and the location of the prefix servers should be chosen so as to efficiently divide the workload between central suffix server and prefix servers.

In our prefix caching assisted periodic broadcast model, we choose to transmit the prefix via *controlled multicast*, while the suffix is delivered through *tailored periodic broadcast*.

The Distribution Network We assume that the distribution network is organized as an *overlay* network. An overlay network consists of a collection of nodes placed at strategic locations in existing network, i.e. the Internet. Overlay networks provide the necessary flexibility to realize enhanced services such as

[5] The term broadcast is commonly used in the literature. In our model, broadcast really refers to multicast as the data that are sent only over links that reach clients who are interested in receiving the video.

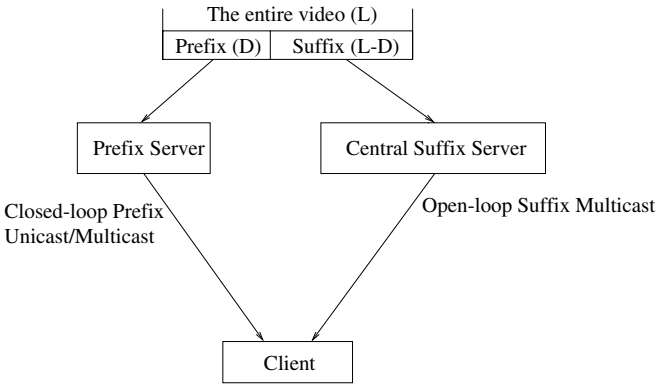


Fig. 1. The VoD distribution architecture

multicast [31] or content distribution [1] and are typically organized in a hierarchical manner.

In our model, we assume that the topology of our distribution network is a m-ary tree with l levels (see figure 2). The suffix server is assumed to be at the root. The prefix servers may be placed at any level of the distribution network other than the highest level (i.e. leaves). If the prefix servers are placed at level j, there will be one at each node of that level, which makes a total of m^j prefix servers. The clients are lumped together at the m^l leaf nodes. The number of clients watching simultaneously a video is not limited to m^l since a leaf node does not represent a single client but multiple clients that are in the same building.

We digress briefly to consider the practical aspects of a network tree model. A tree model captures the hierarchical structure of a large-scale network, where large backbone routers service many smaller service providers which in turn service the end-user clients. For example, a tree might include multiple levels, dividing the network into national, regional and local sub-networks.

The distribution network is assumed to support both unicast and multicast transmissions. Unicast transmission occurs between a server and a single client, whereas multicast transmission occurs when multiple clients (possibly from different leaf nodes) all simultaneously receive the same single transmission from a server. We assume that for the duration of a transmission, a cost must be paid only for every link spanned between the server and its active client(s). The per-link cost may differ depending upon the specific links that are utilized.

For a multicast transmission, the cost may change over the duration of the transmission as users join and leave. Note also that if multiple clients reside at a single leaf node then the cost of multicast transmission is effectively the same as if there were only a single client at that node. For clients at different nodes, multicast still offers savings due to links shared at the lower levels by different nodes.

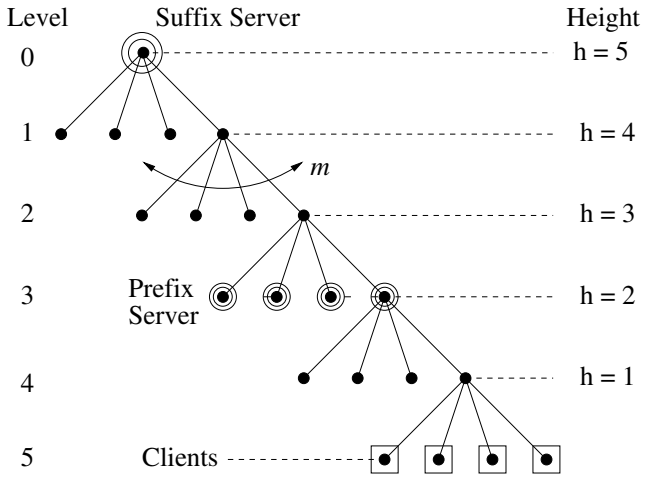


Fig. 2. Video distribution network

Prefix Transmission via Controlled Multicast Patching was first proposed in [26] and then extended with the inclusion of a thresholding policy to produce **Controlled Multicast** [19]. The key idea of patching is to allow clients to share segments of a video stream when they arrive at different times. As the number of clients increases from one to several, the transmission stream is changed from a unicast stream to a multicast one so that late arrivals can still share in the remainder of the stream. In addition, a separate unicast stream must also be transmitted to each client after the first one in order to deliver the data missed due to its later arrival.

For extremely late arrivals, the cost of the additional unicast transmission may outweigh the benefits of sharing in the remaining transmission. Controlled multicast modifies patching to allow for this scenario. Whenever a new transmission is started at time t, arriving clients are patched onto the stream until time $t+T$, where T is a **thresholding** parameter. The first client to arrive after time $t+T$ is given a brand new transmission, and all future arrivals are patched onto the new transmission instead of the old one, until the threshold time passes again and the process is repeated. Figure 3 illustrates the operation of controlled multicast. At time t_1 the first client arrives: The prefix server starts transmitting the prefix via unicast. When the second client joins at time t_2, the remaining part of the prefix is multicast to clients 1 and 2. Client 2 additionally receives the initial part of the prefix that had been transmitted between t_1 and t_2 via a separate unicast transmission. Since client 3 arrives at time t_3, with $t_3 - t_1 > T$, the prefix server starts a new unicast transmission of the entire prefix.

The costs of controlled multicast have been shown to increase sub-linearly with the arrival rate of requests and the length of the prefix [37]; however, the analysis assumes a network of a single link between the server and all its clients. This is the case when the prefix servers are located one level above the clients.

We refer to that case as **leaf prefix server placement**. Placing the prefix servers higher up in the distribution network increases the cost of transmission; however, it consolidates the arrivals to a smaller number of prefix servers and thereby allows for more sharing to occur amongst clients. Furthermore, placing the prefix servers higher up in the network reduces server costs since there are fewer copies of the prefix. One contribution of this paper is an analysis of the tradeoffs in prefix server placement for controlled multicast.

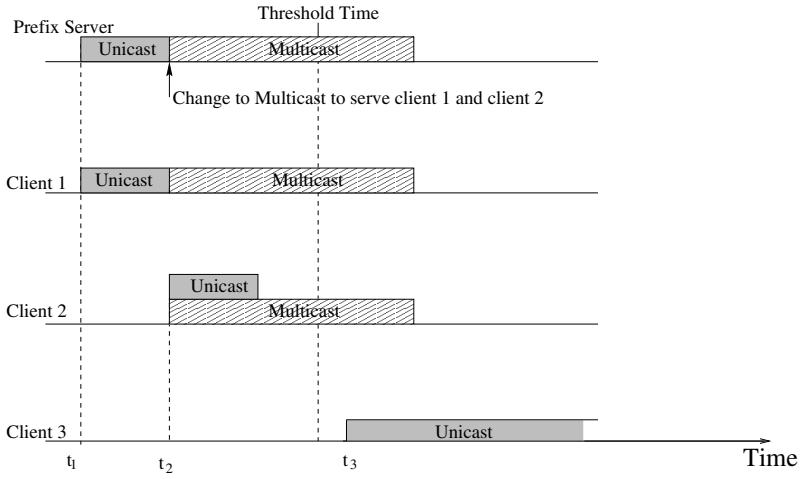


Fig. 3. Prefix transmission with multicast and patching

Tailored Periodic Broadcast of the Suffix We use tailored transmission [11] to transmit the suffix. Thereby, we divide the suffix into segments of fixed lengths. If there are no clients then the suffix server does not transmit. As long as there is at least one client, each segment is periodically multicast at its own transmission rate. Arriving clients receive the multicast of each segment simultaneously. Clients are not expected to arrive at the starting point of each segment; instead, they begin recording at whatever point they arrive, store the data and reconstruct each segment as they receive the data.

The length and rate of each segment is chosen so as to minimize the bandwidth subject to the constraint that each segment must be completely received before its time of playback, and also subject to the storage and reception capabilities of the clients. Figure 4 illustrates the tailored transmission scheme for the case of minimal transmission rates. The client starts receiving all segments at time t_0. The shaded areas for each segment contain exactly the content of that segment as received by the client who started recording at time t_0.

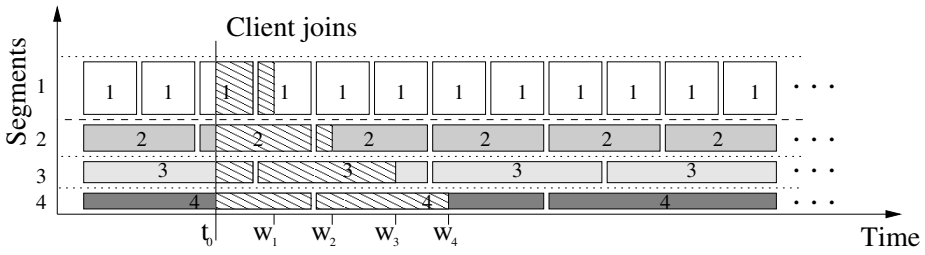


Fig. 4. Tailored Broadcast transmission

The total server transmission bandwidth is:

$$R_t^{min} = \sum_{i=1}^{N_s} r_i^{min}$$

where $N_s \triangleq \lceil \frac{L-D}{D} \rceil$ is the number of segments the suffix consists of and r_i^{min} is the minimal transmission rate of segment i. When we break the suffix of length $L - D$ into segments, each of the segments 1 to $N_s - 1$ has length D and the last segment N_s has a length of $(L - D) - (N_s - 1)D = L - N_sD$. Segment i with length D needs to be entirely received no later than $iD \cdot 60$ seconds after the start of the video consumption. The factor 60 in $iD \cdot 60$ is due to the fact that the lengths of the video, the prefix, and the suffix are expressed in *minutes*. Therefore, the minimal transmission rate for segment i is $r_i^{min} = b\ \frac{D \cdot 60}{iD \cdot 60} = \frac{b}{i}$, where b [Mbit/sec] is the consumption rate of the video and $bD \cdot 60$ is the amount of data [Mbit] in a segment of length D.

In the case where the length of the last segment N_s is less than D (i.e. $\frac{L-D}{D}$ is not an integer), the transmission rate $r_{N_s}^{min}$ is computed as follows depending on whether N_s is larger than 1 or not.

- If $N_s > 1$, the segment N_s should be entirely received $N_sD \cdot 60$ seconds after the start of the video consumption. As we will see later, we assume that all the segments are multiplexed onto a single multicast channel and the receiver stays tuned into that multicast channel until it has received the last segment. Therefore, clients will receive the first segments multiple times. Thus, in this case where the length of segment N_s is less than D, it might be efficient to reduce the tuning time of clients. This can be done by reducing the transmission time of segment N_s to $(N_s - 1)D \cdot 60$ instead $N_sD \cdot 60$ seconds. For a video with a low demand, reducing the service time of the client to $(N_s - 1)D \cdot 60$ increases the transmission rate r_{N_s} of the last segment; however, the increase is offset by avoiding useless transmissions of some of the first segments. For a popular or very popular video, the prefix length is quite short which means that $N_s \sim (N_s - 1)$, and so the increase in r_{N_s} has a negligible impact on the overall server transmission bandwidth R_t^{min}. So, using the fact that $N_s = \lceil \frac{L-D}{D} \rceil = \lfloor \frac{L}{D} \rfloor$, the transmission rate of segment N_s then becomes $r_{N_s}^{min} = b\ \frac{(L-N_sD) \cdot 60}{(N_s-1)D \cdot 60} = b\ \frac{(\frac{L}{D} - \lfloor \frac{L}{D} \rfloor)D}{(N_s-1)D} = b\ \frac{\frac{L}{D} - \lfloor \frac{L}{D} \rfloor}{N_s-1}$.

– If $N_s = 1$, the suffix consists of one segment of length $(L - D) < D$ that must be entirely received $D \cdot 60$ seconds after the start of the video consumption. The transmission rate of segment N_s then becomes $r_{N_s}^{min} = r_1^{min} = b \, \frac{(L-D)\cdot 60}{D \cdot 60} = b \, \frac{L-D}{D}$.

The total server transmission bandwidth is therefore:

$$R_t^{min} = \begin{cases} b \, (\sum_{i=1}^{\lfloor \frac{L-D}{D} \rfloor} \frac{1}{i} + \frac{\frac{L}{D} - \lfloor \frac{L}{D} \rfloor}{N_s - 1}) & \text{if } N_s > 1 \\ b \, \frac{L-D}{D} & \text{if } N_s = 1 \end{cases}$$

Transmitting all the segments at their minimal transmission rate requires that the client starts receiving all segments simultaneously. As a consequence, parts of the video will be received some time before they are consumed and must be stored by the client in the meantime. Figure 5 plots the evolution with time of the amount of data stored for a video of length $L = 90$ min and a prefix length of $D = 2$ min. We see that at the beginning, more and more data will be received ahead of time so that the amount of data keeps increasing until it reaches a peak corresponding to nearly 40 percent of the video. From there on, data will be consumed at a rate higher than the aggregate rate at which new data are received and the storage curve decreases. Storing up to 40 percent of the video data does not pose any problem with todays equipment. In fact, there already exist products such as the digital video recorder by TiVo [40] that can store up to 60 hours of MPEG II encoded video.

The tailored periodic broadcast scheme also allows to support user interactions [10] such as fast forward if the transmission rate of each segment is increased by a small factor (less than twice the minimal transmission rate), provided that the client has enough buffer to store large parts of the video.

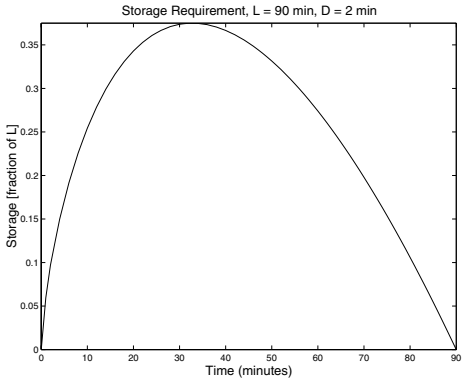


Fig. 5. Storage required at client as function of time, $L = 90, D = 2$

Interaction between Clients and Servers When a new client wants to receive a video, it will contact its closest prefix server in the distribution tree. The prefix server will either start a new transmission cycle of the prefix or extend the ongoing prefix multicast transmission to the new client and transmit a unicast patch to the client. However, the client does not need to contact the suffix server. The suffix server simply needs to know whether there is currently at least one client who needs to receive the suffix, which the suffix server can learn by communicating with the prefix servers. Other large scale video distribution schemes such as hierarchical stream merging [18] require that all client requests be handled by the central server, which makes the central server a potential bottleneck. Our video distribution scheme is not only *scalable* in terms of the network and server resources required to stream a video to a large number of clients but also in terms of processing incoming client requests.

2.3 PS-model, a Cost Model for the Video Transmission

Introduction We divide the costs of a VOD network into network and server costs. The network costs are proportional to the amount of network bandwidth which is transmitted over each link between a server and its clients. The server cost is dependent upon the necessary storage and upon the total number of input/output streams which the server(s) must simultaneously support over the network.

It is interesting to note that the storage and input/output stream capacity of a server cannot be purchased in arbitrary quantities. Instead, they can usually only be purchased in discrete increments. As a result, it is unlikely that a server can be purchased to exactly match both storage and streaming requirements; typically one constraint will be slack as the server will either have extra storage or extra streaming capability.

We will examine the expected delivery cost for a video of length L as a function of the average request rate per *minute* λ (1/min), the prefix length (and allowable suffix delay) D and the topology of the network. The maximum output capacities should be fixed for each server; however, to facilitate the analysis we will assume that any number of streams can be allocated with the costs paid on a per-stream basis.

We divide the multicast tree into levels $1, \ldots, l$, where level 1 consists of the m links from the root and level l consists of the m^l links connected to the leaf nodes. Arrivals to a link at level j are modeled as a Poisson process with parameter λ/m^j. As a consequence, arrivals at the root will form a Poisson process with parameter λ.

We first derive the cost of prefix transmission with a single prefix server at the root. We next generalize the root case to the general case where the prefix servers are placed at some level between the root and the clients. The costs fall into three categories: network, storage capacity and I/O capacity.

We neglect the effects of network latency, even though they can be important from an operational point of view. One might treat latency effects by constraining the maximum number of hops allowed between prefix servers and their clients.

We will derive the cost for the prefix and for the suffix transmission separately and then combine both to obtain the overall system cost. We will refer to this cost model also as the **PS-model** to distinguish it from other cost models.

Costs for Prefix Transmission

Prefix server at the root We first consider a single prefix server at the root of the multicast tree. We will afterwards generalize our results to the case where the prefix server is at an arbitrary height in the tree.

Network bandwidth costs:

A single server at the root combines many small request arrival streams (to the leaves) into a single large arrival stream (to the root); this should lower costs since the cost of prefix transmission via controlled multicast is sub-linear in the arrival rate because it promotes efficiency through shared streams; by combining all the requests arriving at the root, we increase the possibility for sharing. However, this is counterbalanced by the fact that sharing is no longer free; if two clients share the same I/O stream but reside on different leaves, a separate network cost must be paid for each of them. Of course, if clients are already active at every leaf node, then no new network costs must be paid for any future arrivals. However, this scenario is unlikely even for high arrival rates because high arrival rates produce short threshold times in order to reduce the length of the unicast streams.

Let t_i be the time of the ith complete multicast transmission of the prefix without any patching. Arrivals between times t_i and t_{i+1} will share from the multicast transmission at time t_i and will each receive a separate unicast transmission for the data which were missed. We can divide the patching process up into separate renewal cycles $(t_1t_2], (t_2t_3], \ldots$ which are independents and identically distributed in their usage of bandwidth. We analyze the bandwidth usage over a single renewal cycle.

Given the threshold time T, on average there will be $T\lambda$ arrivals which will each need partial transmission of the prefix in unicast. The average length of the unicast transfer will be $T/2$ since the arrivals are uniformly distributed over time. Finally a bandwidth cost must be paid for every link on the path between client (at the leaf) and the root server. As a result, the total amount of data transmitted for the unicast streams over one renewal cycle is

$$C_{netw}^{unicast} = b \cdot 60 \, \frac{lT^2\lambda}{2}.$$

Each arrival will also share from a single multicast network stream. A price must be paid for every link in use. Given a link at level j, let τ_j be the duration of time in which the link is active. For the multicast stream, a link is active from the time of the first arrival (before time T) to that link to the end of the prefix at time D. Arrivals to a link at level j form a Poisson process with parameter λ/m^j.

As each renewal cycle begins with the activation of a stream between the root and a single client, we know that one link at each level will be active at

time zero. Therefore $\tau_j = D$ with probability $1/m^j$. We will now write out an expression for $\mathbb{E}[\tau_j]$:

$$\begin{aligned}\mathbb{E}[\tau_j] &= D\mathbb{P}\{\tau_j = D\} + \mathbb{E}[\tau_j|\tau_j \neq D]\mathbb{P}\{\tau_j \neq D\} \\ &= D\frac{1}{m^j} + \mathbb{E}[\tau_j|\tau_j \neq D]\frac{m^j - 1}{m^j}.\end{aligned}$$

Given a Poisson process with parameter λ/m^j, the time of first arrival will have an exponential distribution with parameter λ/m^j and a cumulative distribution $F(t) = 1 - e^{-\frac{\lambda}{m^j}t}$. We evaluate $\mathbb{E}[\tau_j|\tau_j \neq D]$ making use of the fact that $\int_0^T t\frac{\lambda}{m^j}e^{-\frac{\lambda}{m^j}t}dt = \int_0^T (e^{-\frac{\lambda}{m^j}t} - e^{-\frac{\lambda}{m^j}T})dt$.

$$\begin{aligned}\mathbb{E}[\tau_j|\tau_j \neq D] &= \int_0^T (D-t)\frac{\lambda}{m^j}e^{-\frac{\lambda}{m^j}t}dt \\ &= \int_0^T D\frac{\lambda}{m^j}e^{-\frac{\lambda}{m^j}t}dt - \int_0^T t\frac{\lambda}{m^j}e^{-\frac{\lambda}{m^j}t}dt \\ &= D(1 - e^{-\frac{\lambda}{m^j}T}) - \int_0^T (e^{-\frac{\lambda}{m^j}t} - e^{-\frac{\lambda}{m^j}T})dt \\ &= D(1 - e^{-\frac{\lambda}{m^j}T}) - (-\frac{m^j}{\lambda}e^{-\frac{\lambda}{m^j}t} - e^{-\frac{\lambda}{m^j}T}t)\Big|_{t=0}^{T} \\ &= D(1 - e^{-\frac{\lambda}{m^j}T}) - \frac{m^j}{\lambda} + \frac{m^j}{\lambda}e^{-\frac{\lambda}{m^j}T} + Te^{-\frac{\lambda}{m^j}T}.\end{aligned}$$

Substituting $\mathbb{E}[\tau_j|\tau_j \neq D]$ into $\mathbb{E}[\tau_j]$ produces

$$\begin{aligned}\mathbb{E}[\tau_j] &= D\frac{1}{m^j} + \mathbb{E}[\tau_j|\tau_j \neq D]\frac{m^j - 1}{m^j} \\ &= D\frac{1}{m^j} - \left(\frac{m^j}{\lambda} - \frac{m^j}{\lambda}e^{-\frac{\lambda}{m^j}T} - Te^{-\frac{\lambda}{m^j}T} - D(1 - e^{-\frac{\lambda}{m^j}T})\right)\frac{m^j - 1}{m^j} \\ &= D\left(\frac{1}{m^j} + (1 - e^{-\frac{\lambda}{m^j}T})\frac{m^j - 1}{m^j}\right) - (1 - e^{-\frac{\lambda}{m^j}T})\frac{m^j - 1}{\lambda} + (\frac{m^j - 1}{m^j})Te^{-\frac{\lambda}{m^j}T} \\ &= D(1 - e^{-\frac{\lambda}{m^j}T}(1 - \frac{1}{m^j})) - (1 - e^{-\frac{\lambda}{m^j}T})\frac{m^j - 1}{\lambda} + (\frac{m^j - 1}{m^j})Te^{-\frac{\lambda}{m^j}T}.\end{aligned}$$

By summing over all the links in the tree we find the total multicast cost $C_{netw}^{multicast}$:

$$C_{netw}^{multicast} = b \cdot 60 \sum_{j=1}^{l} m^j \mathbb{E}[\tau_j]$$

and the average network bandwidth cost C_{netw}^{root} can be found by dividing by the average duration of each renewal cycle $(T + 1/\lambda) \cdot 60$.

$$C_{netw}^{root} = \frac{C_{netw}^{multicast} + C_{netw}^{unicast}}{(T + 1/\lambda) \cdot 60} = b \, \frac{\sum_{j=1}^{l} m^j \mathbb{E}[\tau_j] + lT^2\lambda/2}{T + 1/\lambda}.$$

Server costs for prefix server placed at the root:

It is easy to see that the storage cost C_{sto}^{root} of a single root server will be $b{\cdot}60D$. The I/O stream cost must be paid for the output capabilities of each server, i.e. the number of input/output streams which a server can simultaneously maintain. From the expression of C_{netw}^{root} above, one can conclude that the average number of streams for a root server is equivalent to

$$C_{I/O}^{root} = b \, \frac{D + T^2\lambda/2}{T + 1/\lambda}.$$

If T is chosen to minimize the number of I/O streams then

$$C_{I/O}^{root} = b(\sqrt{2D\lambda + 1} - 1).$$

However, choosing T to minimize the number of concurrent I/O streams will unnecessarily increase the network bandwidth. If T is chosen to minimize the network bandwidth, then $C_{I/O}^{root} = b(\sqrt{2CD\lambda/l + 1} - 1 - \frac{D\lambda(C/l-1)}{\sqrt{2CD\lambda/l+1}})$, where C is a scalar constant.

In case of a non-popular video that has on average, at most, one arrival in an interval of time D ($\lambda < 1/D$), each request activates a new prefix transmission and then a new cycle. Hence, the threshold time T becomes *zero* and the expressions for C_{netw}^{root} and $C_{I/O}^{root}$ simplify to

$$C_{netw}^{root} = bl\lambda D$$
$$C_{I/O}^{root} = b\lambda D \ .$$

Varying the placement of the prefix server We now generalize the model to the case where the prefix servers can be placed at any height in the network tree. By placing the prefix servers at some height h in the tree where $1 < h < l$, we divide the arrival process between m^{l-h} servers, each of which can be considered as the root of a network tree with height h. We need only therefore to consider the root server case for a tree of the proper height h with arrival rate λ/m^{l-h}, and then multiply the costs by the number of servers m^{l-h}. The resulting formulas are listed in table 1. The height $h = 1$ refers to the case of a leaf server, i.e. one level before the clients.

Cost terms	
C_{netw}^{prefix}	$b \cdot m^{l-h} \frac{\frac{hT^2\lambda}{2m^{l-h}} + \sum_{j=1}^{h} m^j \mathbb{E}[\tau_j]}{T + m^{l-h}/\lambda}$ $(\mathbb{E}[\tau_j] = D(1 - e^{-\frac{\lambda}{m^j m^{l-h}}T}(1 - \frac{1}{m^j})) - (1 - e^{-\frac{\lambda}{m^j m^{l-h}}T}) \frac{m^{l-h}(m^j-1)}{\lambda}$ $+ (\frac{m^j-1}{m^j}) T e^{-\frac{\lambda}{m^j m^{l-h}}T})$
C_{sto}^{prefix}	$b \cdot 60 \cdot m^{l-h} \cdot D$
$C_{I/O}^{prefix}$	$b \cdot \frac{D + T^2\lambda/2}{T + 1/\lambda}$

Table 1. Summary of the prefix cost terms for the PS-model (l levels, prefix servers at height h)

Costs for Suffix Transmission with a Multicast Tree The costs once again fall into three categories: bandwidth, storage capacity and streaming capacity. The bandwidth cost is equal to the transmission rate R_t^{min} multiplied by the average number of active links, which we will now calculate. For the periodic broadcast, each arrival is serviced for an amount of time $\mathbb{E}[T_s]$ (min), which equals the transmission time of the last segment:

$$\mathbb{E}[T_s] = \begin{cases} \lfloor \frac{L-D}{D} \rfloor \cdot D & \text{if } N_s > 1 \\ D & \text{if } N_s = 1 \end{cases}$$

We assume that all segments are multiplexed to a single multicast channel. As a consequence, each client will consume a bandwidth of R_t^{min} during all the transmission of the suffix. If one multicast channel were dedicated to each segment, the bandwidth consumption could be reduced; the client would be connected only to channels corresponding to segments not yet received. However, this reduction in bandwidth cost comes at the expense of a more complex multicast transmission and a complex synchronization between channels. This study is left for future work.

From queuing theory, it can be shown that given an expected service time $\mathbb{E}[T_s]$ and memoryless arrivals with parameter $\frac{\lambda}{m^j}$, the probability of n jobs simultaneously in progress is given by

$$\mathbb{P}\{n \text{ jobs}\} = \frac{e^{-\frac{\lambda}{m^j}\mathbb{E}[T_s]} (\frac{\lambda}{m^j}\mathbb{E}[T_s])^n}{n!},$$

which is a Poisson distribution. This result can be found through the derivation of the Erlang call-blocking formula commonly used in telecommunications. Arrivals to a link at level j are memoryless with parameter λ/m^j. Define $P(j)$ as the probability that a link at level j has any requests:

$$P(j) \triangleq 1 - \mathbb{P}\{0 \text{ jobs}\} = 1 - e^{-\frac{\lambda}{m^j}\mathbb{E}[T_s]}$$

Then

$$P(j) = \begin{cases} 1 - e^{-\frac{\lambda \lfloor \frac{L-D}{D} \rfloor D}{m^j}} & \text{if } N_s > 1 \\ 1 - e^{-\frac{\lambda D}{m^j}} & \text{if } N_s = 1 \end{cases}$$

The expected number of active links at any given time is therefore

$$\mathbb{E}[\text{active links}] = \sum_{j=1}^{l} m^j P(j),$$

and the bandwidth is

$$C_{netw}^{suffix} = R_t^{min} \sum_{j=1}^{l} m^j P(j). \tag{1}$$

On the other hand, the suffix is continuously and periodically multicast independent of the arrival rate as long as there is at least one user (if there are no users, the suffix server will not send the suffix). In contrast, the number of I/O channels does vary with L and D. The I/O stream cost is equal to the rate $R_t^{min} \times P(0)$, where $P(0)$ is the probability that there is at least one user active at the root:

$$C_{I/O}^{suffix} = \begin{cases} b \left(\sum_{i=1}^{\lfloor \frac{L-D}{D} \rfloor} \frac{1}{i} + \frac{\frac{L}{D} - \lfloor \frac{L}{D} \rfloor}{N_s - 1} \right)(1 - e^{-\lambda \lfloor \frac{L-D}{D} \rfloor D)}) & \text{if } N_s > 1 \\ b \, \frac{L-D}{D}(1 - e^{-\lambda D}) & \text{if } N_s = 1 \end{cases}$$

The storage cost is proportional to the length of the suffix, which is $C_{sto}^{suffix} = b \cdot 60(L - D)$.

The terms of the suffix costs are given in table 2, while table 3 shows all the important mathematical terms used.

Overall System Cost for the Case of a Single Video We divide the costs of a VoD service into network and server costs. The network costs are proportional to the amount of network bandwidth that is expended for the transmission of the prefix and the suffix. The server costs depend upon the necessary storage and upon the total number of input/output streams needed for the suffix server and the prefix server(s).

The total cost of the system can be computed as the sum of the total network and total server costs:

$$C_{PS}^{system} = C_{netw}^{system} + \gamma C_{server}^{system} \tag{2}$$

To relate the network and the server costs, a normalization factor γ is introduced that allows us to explore various scenarios for the cost of the servers as compared

Cost terms	
C_{netw}^{suffix}	$b\ (\sum_{i=1}^{\lfloor\frac{L-D}{D}\rfloor} \frac{1}{i} + \frac{\frac{L}{D} - \lfloor\frac{L}{D}\rfloor}{N_s - 1}) \sum_{j=1}^{l} m^j (1 - e^{-\frac{\lambda \lfloor\frac{L-D}{D}\rfloor D}{m^j}})$ if $N_s > 1$
	$b\ \frac{L-D}{D} \sum_{j=1}^{l} m^j (1 - e^{-\frac{\lambda D}{m^j}})$ if $N_s = 1$
C_{sto}^{suffix}	$b \cdot 60\ (L - D)$
$C_{I/O}^{suffix}$	$b\ (\sum_{i=1}^{\lfloor\frac{L-D}{D}\rfloor} \frac{1}{i} + \frac{\frac{L}{D} - \lfloor\frac{L}{D}\rfloor}{N_s - 1})(1 - e^{-\lambda \lfloor\frac{L-D}{D}\rfloor D})$ if $N_s > 1$
	$b\ \frac{L-D}{D}(1 - e^{-\lambda D})$ if $N_s = 1$

Table 2. Summary of the suffix cost terms for the PS-model.

Term	**Definition**
m	Tree breadth
l	Tree depth
h	Prefix server height
L	Video length (min)
D	Prefix length (min)
$L - D$	Suffix length (min)
N_s	Number of segments of the suffix ($N_s \triangleq \lceil\frac{L-D}{D}\rceil$)
λ	Average client request arrival rate (1/min)
N	Video popularity ($N \triangleq \lambda L$)
τ_j	Active occupation duration for link at depth j (prefix transmission)
$P(j)$	Prob. link at depth j is active (suffix transmission)
T	Threshold time
b	Consumption rate of the video [Mbit/sec]
C	Scalar constant

Table 3. Important Mathematical Terms

to the cost for the transmission bandwidth. We consider here the values of $\gamma = \{0, 0.1, 1\}$. The case of $\gamma = 0$ corresponds to the case that only the cost for network transmission is taken into account and the cost for the servers is not

considered at all (considered to be zero). $\gamma = 0.1$ provides high network cost relative to the server cost, while $\gamma = 1$ represents the case where the network cost is relatively low as compared to the server cost.

The terms for the network and server costs are given by:

$$C_{netw}^{system} = C_{netw}^{prefix} + C_{netw}^{suffix}$$
$$C_{server}^{system} = C_{server}^{prefix} + C_{server}^{suffix}$$

The server cost depends on both, the required amount of storage C_{sto} (in Megabit) and the amount of disk I/O bandwidth $C_{I/O}$ (in Megabit/sec).

$$C_{server}^{prefix} = \max(C_{I/O}^{prefix}, \beta C_{sto}^{prefix})$$
$$C_{server}^{suffix} = \max(C_{I/O}^{suffix}, \beta C_{sto}^{suffix})$$

To be able to relate the cost for storage and for I/O, we introduce the normalization factor β that is determined as follows: If our server has a storage capacity of d_{sto} [Megabit] and an I/O bandwidth of $d_{I/O}$ [Megabit/sec], then $\beta \triangleq \frac{d_{I/O}}{d_{sto}}$. Since the server will be either I/O limited (I/O is the bottleneck and no more requests can be served) or storage limited (storage volume is the bottleneck and no more data can be stored), the server cost is given as the *maximum* of $C_{I/O}$ and βC_{sto}.

To model a case where the cost for the "last-hop link" towards the clients is not the same as the cost for the other links, we can set the cost for the last link to the clients (lhc) to a value different from the cost for the other links.

2.4 Results for Long Videos

Introduction We consider a distribution network with an out-degree $m = 4$ and a number of levels $l = 5$. We expect that such a topology is representative for a wide distribution system that covers a large geographical areas of the size of a country such as France or the UK. If one wants to model a densely populated metropolitan area such as NewYork, one would choose $l < 5$ (e.g. $l = 2, 3$) and $m > 4$ (e.g. $m = 10$). Our model has quite a few parameters and we present results only for a limited subset of parameter values that can provide new insights. For the rest of the paper, we will vary only the parameters γ, last-hop cost lhc, and video length L. The other parameters are chosen as follows: For the disk I/O cost to disk storage cost ratio β, we choose $\beta = 0.001$, which is a realistic value for the current disk technology such as the IBM Ultrastar 72ZX disk. The video length will be $L = 90$ min for most of the time, except when we consider very short video clips of lengths $L = 2$ and 7 min.

Homogeneous Link Costs For the first results presented, the server cost is weighted by $\gamma = 1$ and the network per-link costs are uniform at all levels of the network ($lhc = 1$). We first plot in figure (see figure 6) the optimal system cost as a function of the video popularity N. The video popularity N represents the

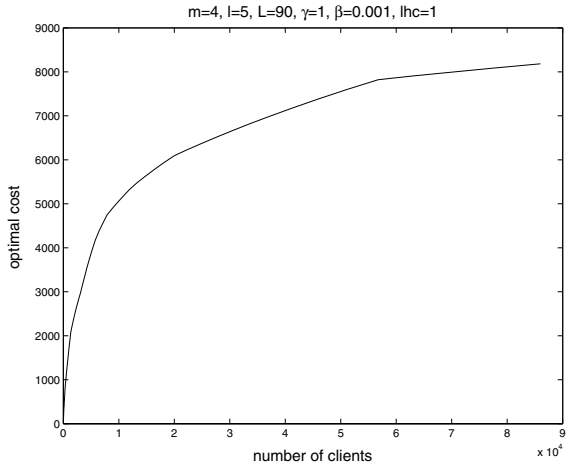


Fig. 6. Total system cost C^{system} with optimal prefix server height and optimal prefix length, for $\gamma = 1$ and $lhc = 1$

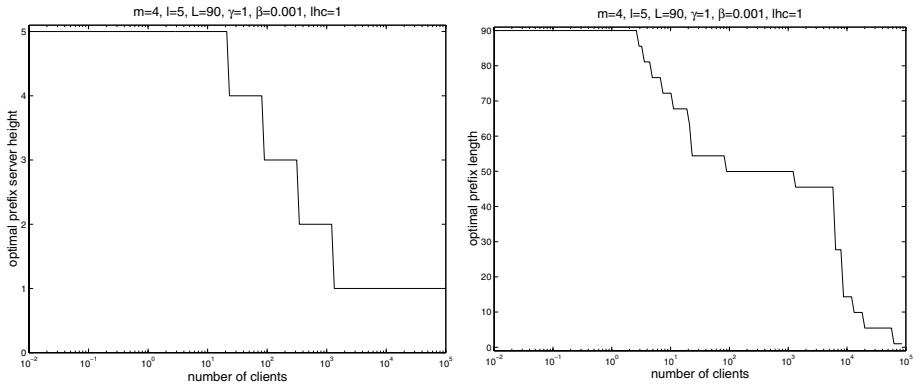
number of clients requesting a same video in an interval of time of duration L ($N = \lambda L$).

In figure 6 we see that the total cost efficiency improves with increasing video popularity N: For a 10-fold increase in video popularity N from 9,000 to 90,000 clients, the cost only doubles.

The optimal values for the prefix length and prefix server placement in the hierarchy as a function of the video popularity N are given in figures 7(a) and 7(b). We see that both, the prefix server height and the prefix length decrease monotonically with N.

For videos that are rarely demanded ($N < 20$), the prefix server is placed at the *root* and the optimal prefix comprises the whole video of 90 min. Indeed, for $N < 20$ the storage cost due to a replication of the prefix in multiple prefix servers is not justified and the optimal architecture is a *centralized* one. On the other hand, for videos that are popular or very popular, the optimal architecture is a *distributed* one with the server for the suffix at the root and the prefix servers closer to the clients. As the popularity N increases, the optimal prefix length decreases since the transmission bandwidth required by the prefix server increases with the square root of the number of clients served, while for the suffix server, the transmission bandwidth required depends for very high values of N only on the length of the suffix and not the number of clients served.

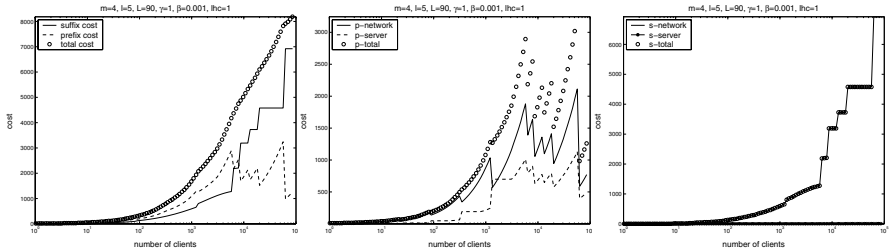
We plot the prefix-suffix cost breakdown in figure 8. We see that the suffix cost C^{suffix} is initially lower than the prefix cost C^{prefix} since the prefix length is quite long. Eventually, for an increasing video popularity N, the suffix system becomes more cost efficient, and so the length of the prefix is significantly reduced and the suffix cost becomes greater than the prefix cost. From figure 8(c) we see that the suffix cost C^{suffix} for higher values of N is entirely determined by



(a) Optimal prefix server height (b) Optimal prefix length (min)

Fig. 7. Optimal prefix server height and optimal prefix length for $\gamma = 1, lhc = 1$.

the suffix *network* cost. In the following, we will not present the suffix cost breakdown anymore since it does not provide any additional insight. For a given suffix length, the fact that C^{suffix} does not change with N indicates that *all* the links of the video distribution network are active, i.e. the multicast transmission is in fact a broadcast to all leaf nodes.



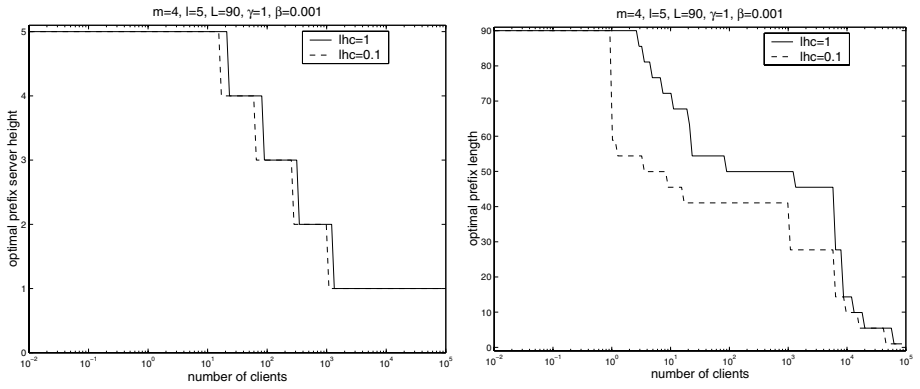
(a) total system cost (b) prefix cost breakdown (c) suffix cost breakdown

Fig. 8. Breakdown of costs for $\gamma = 1$ and $lhc = 1$

If we take a closer look at the evolution of the prefix *server* cost for very popular videos with $N > 10^3$, we see (figure 8(b)) that the prefix server cost increases linearly with increasing N. In this case, to achieve an optimal system cost C^{system}, the prefix length is frequently shortened.

Heterogeneous Link Costs We now set the cost of transmission between levels 4 and 5 (the "last-hop" from the root to the clients at the leaves) to be one-tenth the normal network cost. A reduced last-hop link cost would apply to situations where the local service providers are much cheaper than their regional and national counterparts since they support less traffic. A recent study [8] contains a cost comparison for transmission links of different bandwidth that indicates that the cost in Mbit/hour for local access via ADSL or Cable is at least one order of magnitude lower than the cost for a high speed OC-3 or OC-48 link. We can model this case by using a reduced last-hop link cost, which is $\frac{1}{10}$ of the cost for the other links. We refer to this case as $lhc = 0.1$.

In figure 9, we plot the optimal prefix lengths and prefix server heights under this scenario, with $m = 4, l = 5, \gamma = 1$ and $lhc = \{1, 0.1\}$.



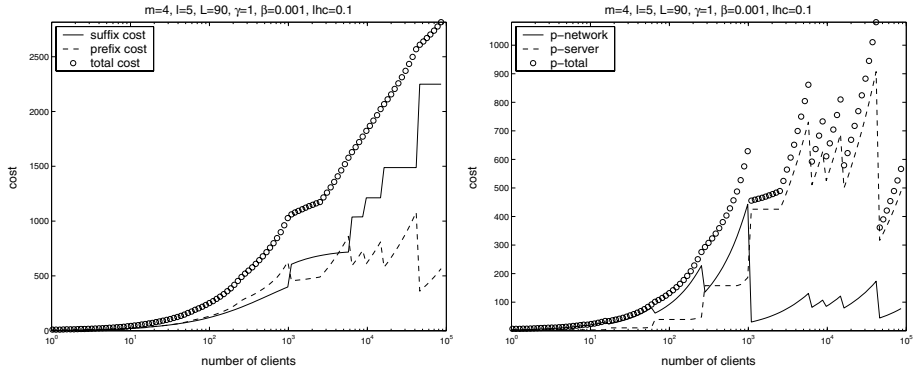
(a) optimal prefix server height (b) optimal prefix length

Fig. 9. Optimal prefix server height and optimal prefix length for $\gamma = 1$ and $lhc = \{1, 0.1\}$

Reducing the last-hop cost makes network transmission cheaper compared to server cost. We see that the prefix server heights are roughly the same while the prefix lengths decay faster than in the original setup ($lhc = 1$) to compensate for the relative increase in prefix server costs. This may seem counter intuitive since reducing the last-hop cost should make the prefix cost cheaper, especially if the prefix servers are at the leaves while the suffix server is at the root. However, we see in figure 10(b) that the *server* cost for the prefix C^{prefix}_{server} now dominates the total prefix cost C^{prefix}, (in particular when the prefix servers are placed at the leaves, i.e. for $N > 10^3$) which was not the case for $lhc = 1$ (see figure 8(b)).

When we compare the suffix costs in figures 8 and 10, we note that reducing the last-hop cost reduces the suffix network cost by almost a factor of 4, which assures that the suffix system remains cost effective. This cost reduction is due

to the fact that with $m = 4, l = 5$, there are 1024 links at the leaf level (last-hop) and only 340 links in the rest of the network. The network cost for the suffix system and the server cost for the prefix system are roughly in the same neighborhood. As a result, the server cost (i.e. the prefix server cost) is magnified in importance and the optimal policy is to compensate by shortening the prefix.



(a) total system cost (b) prefix cost breakdown

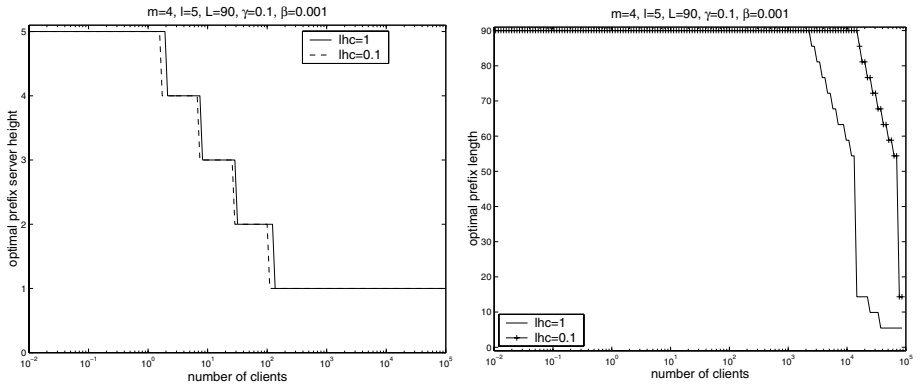
Fig. 10. Breakdown of costs for $\gamma = 1$ and $lhc = 0.1$

Reducing the Server Costs Relative to the Network Transmission Cost While the cost for servers is usually comparable in Europe and the US, the cost of network transmission is much *higher* in Europe than in the US. To account for the case where the network transmission cost relative to the server cost is higher, we set γ to 0.1.

The results for $\gamma = 0.1$ (figure 11) indicate that reduced relative server cost allows to deploy more servers in order to reduce the impact of the expensive network transmission cost. If we compare with $\gamma = 1$ (figure 9), we see that for $\gamma = 0.1$

- The optimal prefix (figure 11(b)) comprises the entire video for a much wider range of clients N. Only for a very high video popularity $N > 10^4$, the optimal prefix length decreases significantly.
- The prefix server height (figure 11(a)) drops faster to $h = 1$ since this helps to reduce the network costs and since the server costs, though they increase (the number of prefix server increases), have been discounted.

We also examine the case where $\gamma = 0.1$ and in addition the network cost for the last-hop is reduced to $lhc = 0.1$. We plot the optimal prefix lengths and prefix



(a) optimal prefix server height (b) optimal prefix length

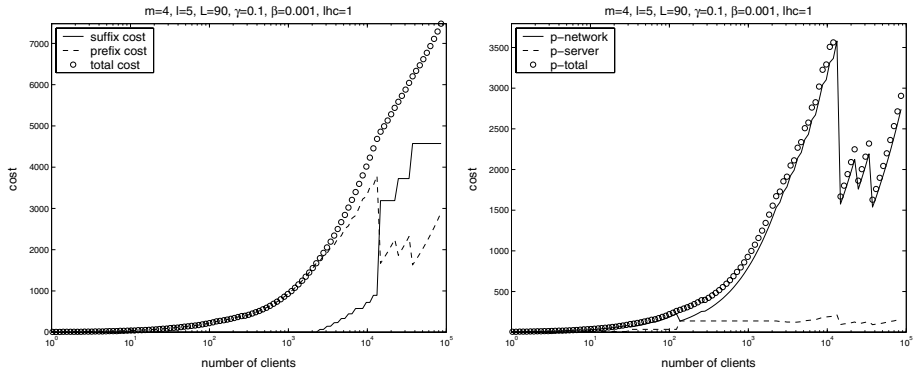
Fig. 11. Optimal prefix server height and optimal prefix length for $\gamma = 0.1$

server positions in figure 11. Previously, we saw for $\gamma = 1, lhc = 0.1$ that although the reduced last-hop cost significantly reduced the cost of prefix service (and of suffix service), the high server costs associated with leaf servers limited the use of the prefix. Now the server cost has been discounted, we see that reducing the last-hop network cost will allow the prefix servers to deliver the entire video over a wider range of video popularities as compared to the $lhc = 1$ case (figure 11). This is interesting, as reducing the last-hop network cost *decreased* the prefix length in the $\gamma = 1$ case and is an example of the interplay between network and server costs. For $\gamma = 1, lhc = 0.1$ the prefix server cost dominates the overall prefix cost (see figure 10(b), while for $\gamma = 0.1, lhc = 0.1$ it is the prefix network cost that dominates the overall prefix cost (see figure 13(b)).

Ignoring Server Costs Previous studies of video distribution schemes [13] have often ignored the server cost and only considered the network cost. We can model this case if we choose $\gamma = 0$. When we ignore the server cost, placing the prefix servers at the leaves is always optimal since the prefix *network* cost is minimized in this case.

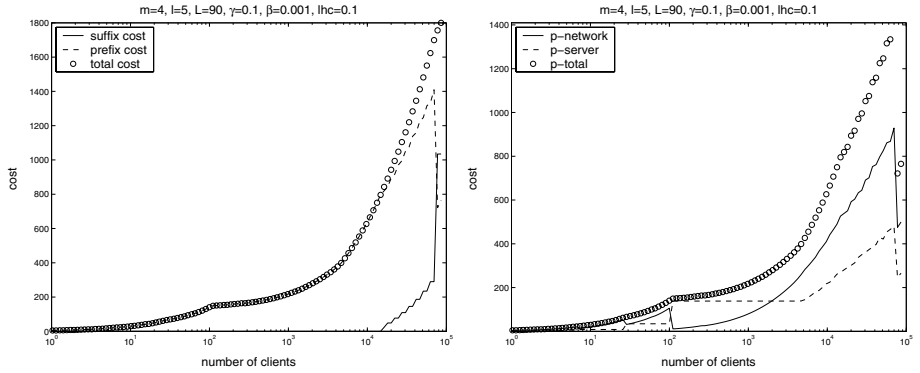
We plot the optimal prefix length for $\gamma = 0$ in figure 14(b). For both values of $lhc = \{1, 0.1\}$, the optimal prefix comprises the entire video for a large interval of the values of N. For large $N > 10^4$, the optimal prefix length becomes shorter as the centralized suffix system becomes more bandwidth efficient (despite the fact that the video must traverse 5 hops for the suffix as compared to 1 hop for the prefix) than the prefix transmission via controlled multicast.

If we compare the optimal values for the prefix length with the case of uniform link costs (i.e. $lhc = 1$) and large N ($N > 10^4$) , we see that for $\gamma = 0$ the optimal values are only unnoticeably larger than for the case of $\gamma = 0.1$.



(a) total system cost (b) prefix cost breakdown

Fig. 12. Breakdown of costs for $\gamma = 0.1$ and $lhc = 1$



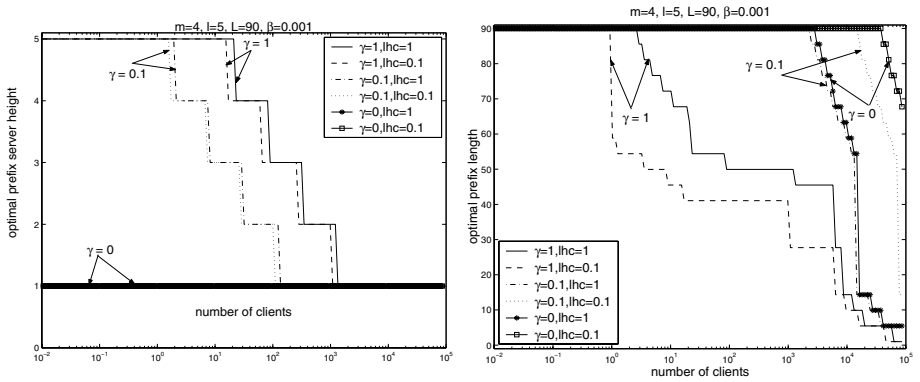
(a) total system cost (b) prefix cost breakdown

Fig. 13. Breakdown of costs for $\gamma = 0.1$ and $lhc = 0.1$

We plot the optimal prefix server height and optimal prefix lengths for all configurations covered so far in figure 14. We see how the system adapts the partitioning into prefix and suffix and the placement of the prefix servers as a function of the cost for the network and server resources. When the cost for the servers relative to the cost for network transmission is reduced ($\gamma < 1$) more prefix servers are deployed. For a given video popularity N, this happens in two ways

- The prefix servers are placed closer to the clients (see figure 14(a))
- The prefix is made longer (see figure 14(b))

In the case of $\gamma = 0$, the entire video is, over a wide range of the video popularity N, delivered by the prefix servers.



(a) optimal prefix server height

(b) optimal prefix length

Fig. 14. Optimal prefix server height and optimal prefix length for $\gamma = \{1, 0.1, 0\}$ and $lhc = \{1, 0.1\}$

We conclude this subsection by showing in figure 15 the optimal total cost values under each scenario discussed. We see that

- for the topology $m = 4, l = 5$ chosen, the majority of the links are at the last-hop to the leaves. Therefore, for non-uniform link costs ($lhc = 0.1$) the cost of the overall system is considerably much smaller than for $lhc = 1$.
- There is surprisingly little difference in the cost of the overall system for $\gamma = 0.1$ and $\gamma = 0$ since in both cases it is the cost for the network transmission that dominates the total cost. For $\gamma = 0.1$, the impact of the prefix *server* cost is attenuated (for $1 < N < 100$) by using fewer prefix servers (see figure 14(a))

Conclusions So Far We have seen how our distribution architecture gives us the cost-optimal configuration as a function of the video popularity N. For a non-popular video, the prefix server is placed at the root[6] and the length of the prefix comprises the whole duration of the video. As the video popularity N increases, the optimal prefix length becomes shorter and the optimal height for prefix servers becomes closer to the clients.

[6] With the exception of $\gamma = 0$, where the prefix servers are always placed at height $h = 1$.

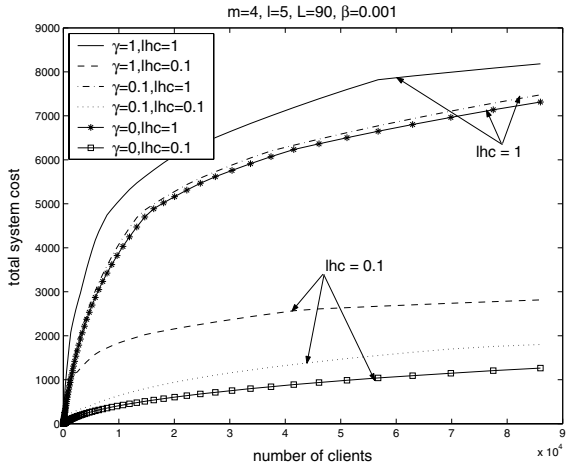


Fig. 15. Comparison of total system costs for $\gamma = \{1, 0.1, 0\}$ and $lhc = \{1, 0.1\}$

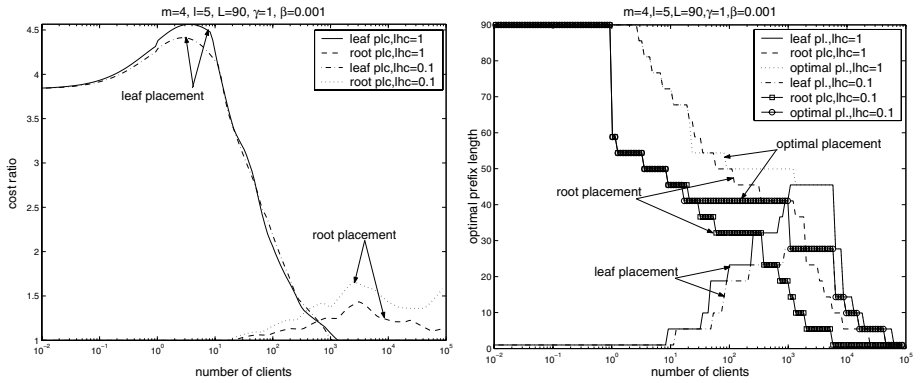
Using the suffix server at the root of the distribution tree becomes economically very interesting for very popular videos, where the prefix length becomes much shorter than the whole duration of the video. However, the optimal suffix length used is quite sensitive to changes in the popularity (see figure 14(b) for $10^4 \leq N \leq 10^5$). Therefore, to operate the distribution system in a cost-optimal fashion, one needs to periodically estimate the current popularity of a video and adapt, if necessary, the prefix length.

Costs for Non-optimal Prefix Server Placement For a large video distribution system serving many movies, it will be feasible to place prefix servers at every level of the distribution tree; however, for smaller systems, the prefix server locations will probably be fixed as there may not be prefix servers at every level of the distribution network. The placement of the prefix servers at an arbitrary level may also be impossible for other reasons. For instance, the facilities necessary for installing a server may not be available or accessible. Thereby, it is worth evaluating the cost performance of a video distribution system where the prefix servers are placed non-optimally. We will examine how the system adjusts the prefix length to find the most cost-efficient solution for the case where

- The prefix server is placed at the root, which will be referred to as **root placement**
- The prefix servers are placed at the leaves (i.e. at height h = 1, one level above the clients), which will be referred to as **leaf placement**

We always assume that the cost for the server is taken into account (i.e. $\gamma \neq 0$) and consider the following scenarios

- Network is cheap, $\gamma = 1$ and $lhc = \{1, 0.1\}$.
- Network is expensive, $\gamma = 0.1$ and $lhc = \{1, 0.1\}$.



(a) Cost ratio for prefix server, $\gamma = 1$, $lhc = \{1, 0.1\}$

(b) Optimal prefix length, $\gamma = 1$, $lhc = \{1, 0.1\}$

Fig. 16. System cost ratio for non-optimal placement to optimal placement system cost and optimal prefix length for $\gamma = 1$ and $lhc = \{1, 0.1\}$.

Network is cheap We first discuss the case with $lhc = 1$, where the per-link network costs are the same across the network. We know from figure 7(a) that for values of $N, N < 20$, the optimal placement of the prefix server is at the root. For increasing values of $N > 20$, the optimal placement finds the best trade-off between network and server cost by moving the prefix servers closer to clients and reducing the length of prefix. When we fix the placement of the prefix server at the root, the only degree of freedom left to minimize the cost is to adjust the prefix length. We see in figure 16(b) that the prefix length for the root placement case decreases more rapidly than when the prefix placement is chosen optimally.

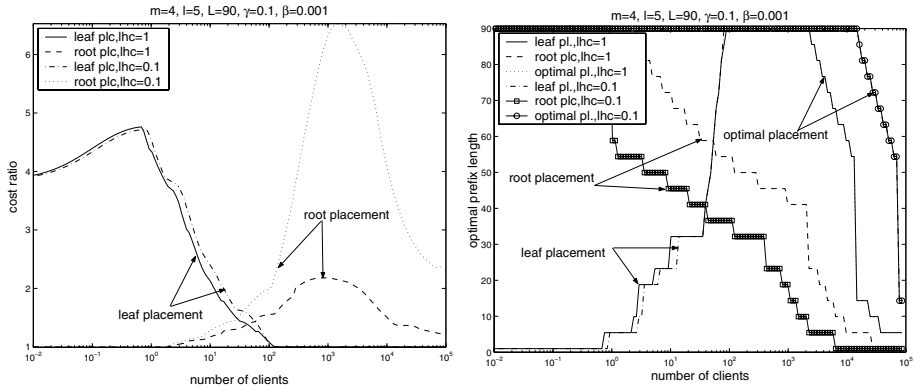
In figure 16(a) we plot the ratio of the total system costs. For the case when the prefix server is always at the root as compared to when it is optimally placed, the total system cost increases by up to 60%: For very popular videos, placing the prefix server at the root results in a longer suffix in order to limit the high cost for the prefix distribution.

When the prefix servers are always at the leaves, the cost increase as compared to the optimal placement is much higher than for root placement and can be up to 450% since leaf placement is very expensive for non-popular videos ($N < 10^2$). This big difference in cost is due to the fact that keeping copies of a video in all the prefix servers placed at the leaves is very inefficient for videos that are not very popular, i.e. $N < 10^3$. When the prefix servers are placed at the leaves and the video popularity is low ($N < 10^1$), the system chooses the minimal possible prefix length of $D = 1$ to limit the overall cost of keeping the many copies of that prefix (see figure 16(b)). For larger values of N, the optimal prefix server placement is at the leaves, and so the performance of the non-optimal system is the same as the optimal one for $N > 10^3$.

When we compare the case where all links of the network have the same cost ($lhc = 1$) to the case where the last-hop links are less expensive for $lhc = 0.1$, we see little difference in the results. For $lhc = 0.1$, the worst case cost increase due to leaf placement is a little bit less than for $lhc = 1$. For the root placement, the fact that the last-hop links are less expensive ($lhc = 0.1$) increases the cost of root placement as compared to $lhc = 1$.

Overall, when the network is cheap and we must choose between root and leaf placement, root placement is preferable.

Network is expensive We examine now the opposite situation, where the network is expensive as compared to the cost for servers (see figure 17(a)).



(a) cost ratio for prefix server, $\gamma = 0.1$, $lhc = \{1, 0.1\}$

(b) Optimal prefix length, $\gamma = 0.1$, $lhc = \{1, 0.1\}$

Fig. 17. System cost ratio for non-optimal placement to optimal placement and optimal prefix length for $\gamma = 0.1$ and $lhc = \{1, 0.1\}$

When the network is expensive, the worst case cost performance of leaf placement deteriorates only slightly as compared to the case when the network is cheap, since the cost for the servers placed at the leaves dominates the total system cost. For root placement, the worst case cost is significantly higher as in the case of $\gamma = 0.1$, in particular for $lhc = 0.1$, since the network transmission has become more expensive, which is directly reflected in an increase of the total system cost. The optimal placement for $\gamma = 0.1$ moves the prefix servers for increasing N rapidly towards the clients and chooses a prefix that comprises the entire video for all except the very popular ones $N > 10^4$ (see figure 14). Since the root placement can not move the prefix servers, it shortens the prefix length drastically with increasing N to offset the cost-increase due to the prefix placement at the root (see figure 17(b)).

Conclusion The video delivery architecture has normally two degrees of freedom: the prefix length and the prefix server placement can be varied to determine the cost optimal solution. When we remove one degree of freedom and fix the placement of the prefix server, the total system cost can increase significantly, in particular for the case of leaf placement, where the server cost dominates as compared to the network cost.

2.5 Short Videos

Introduction So far we have looked at videos of length $L = 90$ min, which corresponds to feature movies. Besides movies, there are news clips or clips for product promotion, which are much shorter in length, that can be distributed via a video distribution system. For these clips it is interesting to evaluate how efficiently the video distribution architecture supports the distribution of these clips. We consider clips of $L = 7$ min[7]. As before, the network has an outdegree $m = 4$ and a number of levels $l = 5$. We vary the popularity of the video between $10^{-2} \leq N \leq 10^5$.

The optimal configuration is computed using the PS-model. The PS-model allows to determine which fraction of the video should be stored at the prefix servers and where to place the prefix servers to minimize the delivery cost.

In addition, we consider two more cases where we remove one degree of freedom:

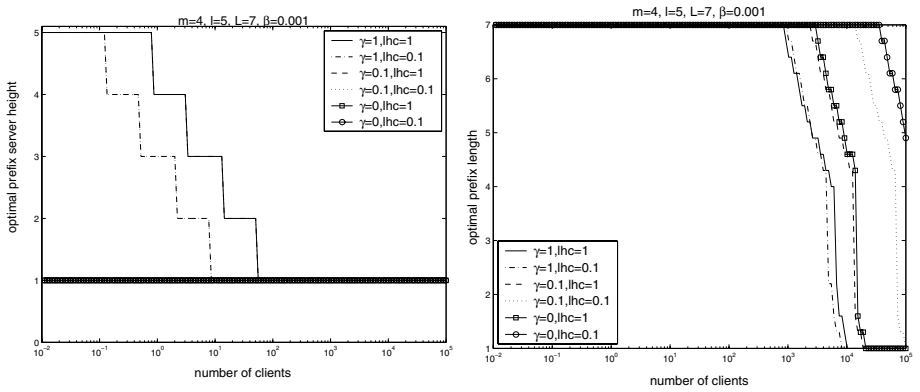
- $D = L$, i.e. the prefix comprises the full video, which we refer to as **full video caching**. However, the height of the prefix servers is chosen such to minimize the overall system cost.
- The prefix server is fixed at the root, which we introduced in subsection 3 as **root placement**. However, the length of the prefix is chosen such to minimize the overall system cost.

Results Figure 18 shows the optimal prefix server placement and the optimal prefix length in the hierarchy as a function of the video popularity N. A comparison with the values obtained for long videos of $L = 90$ min (see figure 14) shows that the video distribution system behaves the same way, for both short and long videos:

- With increasing video popularity N, the placement of the prefix servers moves closer to the clients
- For all but the most popular videos, the optimal prefix comprises the whole video.

As we can observe from figures 18(b) and 19(a), full video caching is the optimal solution, except for the most popular videos.

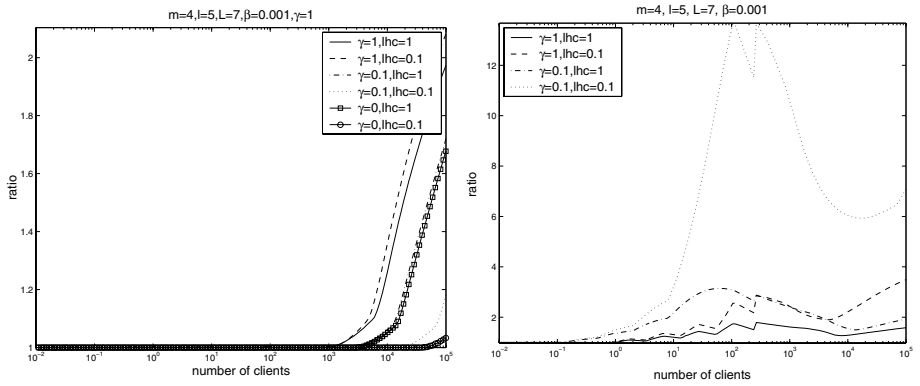
[7] We also looked at clips of 2 min length. The results we have obtained for $L = 2$ are very similar to the ones for $L = 7$ and are therefore not present here.



(a) optimal prefix server height

(b) optimal prefix length

Fig. 18. Optimal prefix server height and prefix length for $\gamma = \{1, 0.1, 0\}$, $lhc = \{1, 0.1\}$, and $L = 7$ min.



(a) Cost ratio for full video caching

(b) Cost ratio for root placement

Fig. 19. Cost ratio for full video caching system and cost ratio of root placement for $\gamma = \{1, 0.1, 0\}$, $lhc = \{1, 0.1\}$, and $L = 7$ min.

In Figure 19(b) we plot the ratio of the delivery cost of a video obtained in the case of root placement as compared to the delivery cost obtained when both, the prefix length and the prefix server placement are chosen optimally[8]. We see

[8] We do not plot the cost ratio for $\gamma = 0$, which can reach a value up to 40 for small values of N.

that there is an additional cost of fixing the prefix server at the root for a values of video popularity N except very small ones, where placing the prefix server at the root is the optimal choice. The additional cost due to root placement is lowest when the network transmission is cheap ($\gamma = 1$) and highest when the relative cost for the prefix servers is low ($\gamma = 0.1$) **and** the transmission cost over the last hop is reduced ($lhc = 0.1$). When the network is expensive ($\gamma = 0.1$) the cost ratio is worst for the values of N where the optimal prefix server placement puts the prefix servers at the leaves ($h = 1$) and chooses a prefix that comprises the entire video ($D = L$). The shape of the curves is similar to the ones observed for long videos (see figures 16(a) and 17(a)).

2.6 Video Distribution System for a Set of Videos

Introduction So far we have looked at a single video. We studied the case of how to determine the optimal prefix length and the optimal prefix placement as a function of the video popularity N. However, a complete video distribution system will offer to the clients a host of different videos $\mathcal{V} = \{1, \ldots, K\}$ to choose from.

We will now extend the PS-model to deal with the following scenarios:

- Provisioning.
  The PS-model is used to solve the provisioning problem for a given set of videos whose popularities are known: We just need to execute the model for each video separately to determine the optimal prefix length and the placement of the prefix servers.
- Video assignment problem for an existing configuration.
  Very often, the situation will be such that the video distribution system has been already deployed, i.e. the central suffix server and the prefix servers have been installed and changing the location (placement) or the capacities of prefix servers is not possible. Given that the placement and the capabilities of the prefix servers are *fixed*, we then want to determine the cost-optimal prefix length and prefix server placement for a set of videos and popularities.

For a set $\mathcal{V} = \{1, \ldots, K\}$ of K videos, the PS-model computes separately the optimal system cost of each video $i \in \mathcal{V}$. The total system cost will be the sum of the system costs over all K videos of the system. In the following, we will consider that the popularity $N_i = \frac{A}{i^\alpha}$ of video $i \in \mathcal{V}$ follows a Zipf distribution the popularity of the most popular video and α the slope of the popularity distribution; the bigger α, steeper the slope, i.e. the more biased or skewed the popularity distribution. In figure 20 we plot the popularity for different values of A and α.

Provisioning of the Prefix Servers The aim of provisioning is to compute the resources required in terms of video server storage and video server I/O bandwidth for a given set of videos $\mathcal{V} = \{1, \ldots, K\}$ with popularities N_i, for

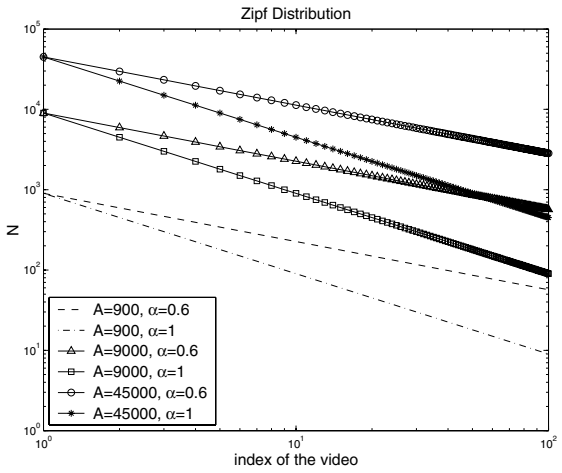


Fig. 20. The popularity N_i of video i as a function of the index i, according to a Zipf distribution, on a log-log scale.

$i \in \mathcal{V}$[9]. We will concentrate here on the provisioning of the prefix servers inside the network. However, the provisioning of the servers at the root can be done in a similar way.

We use equation 2 (subsection 2.3) to compute the optimal prefix length D_i, the optimal threshold T_i, and the optimal prefix server level l_i for each video $i \in \mathcal{V}$. Let $\mathcal{L}(j) \triangleq \{i \in \mathcal{V} \mid l_i = j\}$ denote the subset of videos whose prefix will be optimally served by the prefix servers of level j and I/O_i denote the amount of I/O bandwidth needed for each prefix server at level j to serve the prefix of video $i \in \mathcal{V}$. The value of I/O_i can be computed the same way as $C_{I/O}^{prefix}$ in subsection 2.3. With $\lambda_i = \frac{N_i}{L}$ and $i \in \mathcal{L}(j)$ we have

$$I/O_i = b\ \frac{\lambda_i}{m^j} \frac{2D_i + \lambda_i T_i^2/m^j}{2 + 2\lambda_i T_i/m^j}$$

At each level j, the total storage $PS_{st}(j)$ and I/O $PS_{I/O}(j)$ capacity of the prefix servers are computed as the sum of the prefix lengths and the amount of I/O bandwidth needed over all prefixes placed at that level. Hence, at level j, the resource requirements of the prefix servers are given by:

$$PS_{st}(j) = \sum_{i\ \in\ \mathcal{L}(j)} b \cdot 60\ D_i \qquad \forall j \in \{1, \ldots, l-1\}$$

$$PS_{I/O}(j) = \sum_{i\ \in\ \mathcal{L}(j)} I/O_i \quad \forall j \in \{1, \ldots, l-1\}$$

[9] For sake of simplicity we assume that all videos have the same length L.

Since we assume a homogeneous client population, the load will be uniformly distributed over the prefix servers at a particular level. Therefore, all prefix servers at a particular level j will have the same capabilities.

Assignment of a Videos to Prefix Servers with Limited Storage and I/O Capabilities We now consider the case that the prefix servers have been installed and that it is not possible to add new prefix servers or to modify their placement in the distribution hierarchy. The values of $PS_{st}(j)$ and $PS_{I/O}(j)$ are known $\forall j \in \{1, \ldots, l-1\}$. We will refer to them as **prefix server constraints**. However, we allow for modifications to the servers installed at the *root*. This means that there are no constraints on the resources of the central suffix server or the prefix server at the root ($l = 0$).

To solve the prefix assignment problem for a set of videos $\mathcal{V}$, we need to find the placement that satisfies the prefix server constraints of the system and minimizes the total system cost.

We formulate the assignment problem for a set $\mathcal{V}$ of videos as follows:

$$\begin{cases} \min\limits_{\theta_{ij}}(\sum\limits_{j=0}^{l-1}\sum\limits_{i\in\mathcal{V}} C_{PS}^{system}(i,j) \times \theta_{ij}) & \\ s.t. \quad \sum\limits_{i\in\mathcal{V}} D_{ij} \times \theta_{ij} \leq PS_{st}(j) & 1 \leq j \leq l-1 \\ \quad \sum\limits_{i\in\mathcal{V}} I/O_{ij} \times \theta_{ij} \leq PS_{I/O}(j) & 1 \leq j \leq l-1 \\ \quad \sum\limits_{j=0}^{l-1} \theta_{ij} = 1, & \forall\ i \\ \quad \theta_{ij} \in \{0,1\}, & \forall\ i,j \end{cases}$$

where (i) $C_{PS}^{system}(i,j)$ is the lowest total system cost achievable when placing video i at level j, (ii) D_{ij} is the corresponding optimal prefix length when placing video i at level j, (iii) I/O_{ij} is the amount of I/O bandwidth needed when placing video i at level j, and (iv) θ_{ij} is a binary variable. θ_{ij} is equal to 1 if the prefix of the video i is placed at level j and 0 otherwise. $\sum\limits_{j=0}^{l-1} \theta_{ij} = 1$ indicates that no video prefix can be stored at more than one level in the hierarchy.

Both, the objective function and the constraints are linear functions of the binary variables θ_{ij}. This optimization problem can be resolved using dynamic programming. We use the *XPress-MP* package [15] to solve the assignment problem.

Results Moving a video prefix from an optimal level to a non-optimal one in order to satisfy the constraints increases the delivery cost of the video and consequently the overall system cost. It is interesting to evaluate how the prefix

server constraints will impact the overall system cost of the video distribution system. To this purpose, we compute the overall system cost C_{opt}^{vds} *without* any constraints and the overall system cost C_{constr}^{vds} with prefix server constraints, which are defined as

$$C_{opt}^{vds} = \sum_{i \in \mathcal{V}} C_{PS}^{system}(i)$$

$$C_{constr}^{vds} = \sum_{j=0}^{l-1} \sum_{i \in \mathcal{V}} C_{PS}^{system}(i,j) \times \theta_{ij}$$

We use the cost ratio $C_{constr}^{vds}/C_{opt}^{vds}$ to evaluate how well the video distribution architecture can adapt to changes in the video popularity and the number of videos. We plot in figure 21 the cost ratio for $\alpha = \{0.2, 0.6, 1\}$ and different numbers of videos $K = \{100, 120, 150\}$. We set the normalization constant to $A = 9000$. The prefix server constraints were computed for the case of $K = 100$ videos with $\alpha = 0.6$ for the Zipf distribution. This initial distribution ($\alpha = 0.6$) is skewed or biased enough to fit well with small systems.

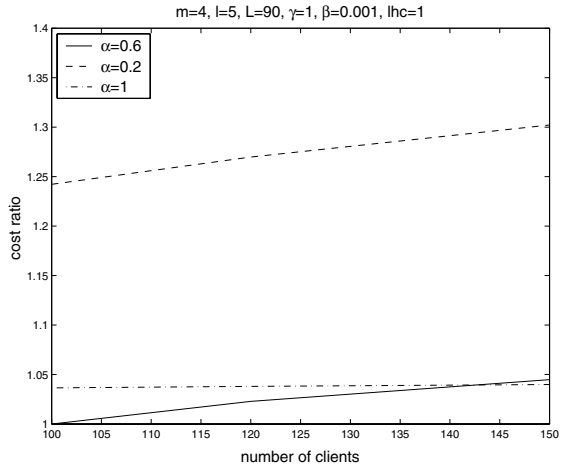


Fig. 21. Cost ratio $C_{constr}^{vds}/C_{opt}^{vds}$.

Figure 21 shows the following interesting features:

- The cost ratio increases sub-linearly with the number of videos.
- The increase of the system cost strongly depends on the parameter α.

These results fit well with the intuition. The larger the popularity N of a video, the closer is the placement of the prefix to the clients. If there is no more place for the popular videos at the higher levels of the prefix hierarchy, video must be moved closer to the root.

For a given Zipf distribution, increasing the number of videos adds videos at the *tail* of the distribution. Those additional videos are the least popular ones, and must be moved near the root to free resources for the more popular ones at the higher levels. The placement of the least popular videos close to the root will use up very few of the constraint resources, which explains the small change in the cost ratio with increasing number of videos. At some point, all the additional videos will have such a low popularity that their prefixes will all be placed at the root and the cost ratio will no longer change with increasing number of videos.

For $\alpha = 1$, the popularity distribution is very skewed, and increasing the number of videos K in the system has no impact on the cost ratio since those additional videos are all optimally placed at the root. If we compare $\alpha = 0.2$ and $\alpha = 0.6$, we find that, as the number of videos K increases, the cost ratio increases more rapidly for $\alpha = 0.2$ than for $\alpha = 0.6$. The reason is that, for $\alpha = 0.2$, the videos added to the system are relatively more popular than those for $\alpha = 0.6$, and as a consequence, moving them closer to (or placing them at) the root to satisfy the constraints comes out more costly.

We also evaluated other scenarios such that, $\gamma = 0.1$, or $\gamma = 0$ and for a reduced cost of the last hop link $lhc = 0.1$. The results obtained were similar and are therefore not presented here.

Conclusion The PS-model adapts itself very well in case of limited prefix server resources. Non-popular videos are moved close to the root to free a place for the more popular ones. For $\alpha = 0.6$ and $\alpha = 1$, the increase in the overall system cost for increasing number of videos is very low, less than 5%. Even when the popularity distribution differs a lot from the one used when determining the prefix server resources (as is the case for $\alpha = 0.2$) the extra cost incurred will be small, 25%– 30%.

2.7 Conclusion and Outlook

Summary We have presented a scalable video distribution architecture that combines open-loop and closed-loop schemes and assure a zero start-up delay. Under this architecture, each video is split into two parts, the prefix and the suffix. The prefix is transmitted via controlled multicast (closed-loop scheme) while the suffix is transmitted via tailored periodic broadcast (open-loop scheme). The architecture is very cost-effective since the cost for the prefix transmission increases only with the square root of the number of clients and the suffix distribution cost is, at high request rates, independent of the number of clients and simply a function of the number of segments the suffix is decomposed into.

Another advantage is that our architecture is highly scalable in terms of serving incoming client requests. A client who wants to receive a video contacts his closest prefix server. The central server only needs to know if there is at least one client connected, which the central server can learn by communicating with the prefix servers. This interaction between the clients and the servers avoids having a bottleneck due to handling all the requests by a single server.

We have developed an analytical cost model for that architecture that we called PS-model. In the cost model, we include not only the network bandwidth cost, but also the costs for the server I/O bandwidth and server storage. Using the PS-model we can determine the

- Bandwidth and streaming costs for prefix and suffix transmissions
- Optimal prefix length
- Optimal position of the prefix servers.

The PS-model makes the trade-off between the server and network bandwidth costs to determine the cost-optimal prefix length and the optimal prefix server placement.

As key results, we found that

- The cost efficiency increases with increasing video popularity N. For a 10-fold increase in N from 9,000 to 90,000 clients the total system cost only doubles.
- A popular video is replicated at many prefix servers that are placed close to the clients.
- A non-popular video is placed at the root or very close to the root to reduce the server storage cost incurred for replicating the prefix across many prefix servers.
- The central suffix server is highly efficient to serve very popular videos for which the optimal the suffix comprises the major portion of the video.

The PS-model has two degrees of freedom: It can adjust the length of the prefix as well as the placement of the prefix server in the hierarchy so as to divide efficiently the workload between the suffix server and the prefix servers. Thus, if we remove one degree of freedom, for instance, we fix the placement of the prefix server at either the root or the leaves, the only degree of freedom left is to adjust the length of the prefix. It is worth to discuss the cost for a non-optimal placement of the prefix servers since for small systems, it might not possible to have prefix servers at any level in the network. By comparing between leaf placement and root placement for the prefix servers, we found that

- The system cost can increase significantly as compared to the optimal system cost. This increase is up to 450% for leaf placement.
- Root placement outperforms leaf placement in the case where the network cost is cheap as compared to the server cost. In this case, the cost increase is relatively small, up to 60%.
- Root placement becomes very costly when the network cost is high relative to the server cost, and the increase in the system cost can exceed 600% in case of a reduced last-hop cost.

We also evaluated the PS-model for the case of short videos such as video clips or clips for product promotions. In fact, even for short videos, it is always cost-optimal to divide the video into prefix and suffix in the case very popular clips.

Moreover, we extended the PS-model to looked at the following scenarios:

- Provisioning:
  We showed how the PS-model can be used to compute the cost-optimal for a system with a set $\mathcal{V}$ of videos. Given the popularity of each video, the PS-model determines the system resources required in terms of network bandwidth, server I/O bandwidth, and server storage to optimize the total system cost.
- Assignment of prefixes into prefix servers:
  We studied how the PS-model adapts the prefix length and the prefix placement for a set $\mathcal{V}$ of videos when the amount of resources for the prefix servers is given, which is for instance the case when a new set of videos must be optimally placed. The cost increase due to predefined capabilities of the prefix servers is very low over a wide region of our model, less than 5%. Even when the popularity distribution of the videos differs a lot from the initial distribution used to determine the prefix server resources, the increase in the system cost remains low, 25-30%.

The PS-model has allowed us to study different scenarios. However, several extensions to the PS-model are possible that allow to make the overall model more realistic. The current model assumes that client requests for a single video are homogeneously distributed among all clients. A possible extension would consider the case where a particular video is more popular with a sub-group of the clients which would lead to heterogeneous request patterns. The current video distribution network has a very regular structure with all clients being at the same distance from the root and the distribution tree being very regular, while a real distribution network most likely has not such a regular structure. We intend to extend our model in the future and evaluate the impact of these extensions. These extensions will clearly change the absolute cost values. However, we do not expect that they will change the broad conclusions that we could draw using the PS-model.

We would also like to evaluate different architectural choices. Today, digital VCRs with at least one hundred Gigabyte of local storage are commercially available [40]. Given local storage, one can proactively download the prefixes of the most popular videos directly into the VCR. We intend to extend the PS-model to evaluate the overall cost reduction due to the use of local storage in the VCR. Another very attractive architectural option is a satellite distribution of the suffix of the videos.

Outlook In the system model we have analyzed, we had assumed that both, the suffix server and the prefix servers, are *dedicated*. These assumptions hold true for a "commercial" CDN. In recent years, a new paradigm called peer-to-peer [34] emerged where a particular machine can assume both roles, i.e. be client and server at the same time. Popular peer-to-peer systems such as Gnutella, Napster, or KaZaa have been used by Millions of users to share digital content such as MP3 files. P2P architectures are very interesting for scalable

content distribution. They offload all or at least the majority of the work for storage and distribution onto end-systems that are *not dedicated* to the purpose of content distribution and therefore, significantly reduce capital expenditures. P2P architectures are also inherently *self-scaling*: In case of a sudden increase in the number of requests, the number of peers that have received the content will also increase, which in turn will increase the total capacity of the P2P system to serve new clients. P2P systems are therefore much more suitable than centralized server-based systems to handle "flash crowds".

The research community has made numerous proposals on how to use the P2P paradigm to provide overlay multicast distribution trees [14, 7] and to perform scalable video distribution. The P^2Cast scheme [22] partitions the video into prefix and suffix as we do in our model and proposes to distribute the suffix by a central server via application layer multicast while the prefix is delivered via unicast by another client that has already received the prefix previously and is currently viewing the suffix.

Constructing application layer multicast trees for video distribution is very challenging as clients that are part of the tree may leave at any time, which may disrupt the video reception of the clients that are downstream while the tree is re-built. Various proposals such as Coopnet [36] and SplitStream [12] propose to encode the video signal using multiple description encoding techniques [20] where the video signal is encoded as N independent descriptions or sub-streams that are each transmitted on a separate multicast tree. In this case, a receiver will be able to play out the video, albeit with reduced resolution, if it receives only a subset of the descriptions. When using such an approach, it is important that the multicast trees for transmitting the different descriptions are constructed in such a way that the failure of a node will not affect different receivers for each of the descriptions. This is assumed in SplitStream by constructing the trees in such a way that an interior node in one tree is a leaf node in all the other trees.

A hybrid architecture that combines a CDN (with its servers installed at certain places) and peer-to-peer based streaming was proposed in [47]. Such an architecture allows to considerably reduce the amount of CDN resources required since peers that have received a video will in turn serve other clients, which reduces that load on the CDN server.

3 Scheduling Objects for Broadcast Systems

3.1 Introduction

Technological progress in high speed communications, embedded systems and consumer electronics has led to the availability of several *thin*, personalized user terminals capable to store, process, transmit and receive information, such as mobile phones, personal digital assistants (PDA), palmtops, tablet PCs, etc. Their powerful characteristics enable a wide range of services and applications, which deliver information to users efficiently.

Broadcast systems constitute a popular communication infrastructure to deliver services to thin, personalized *client* devices. Broadcast systems are capable

to deliver multimedia services to a wide client population, because they are scalable in terms of user population and end-user bandwidth; furthermore, they accommodate heterogeneous user technologies, including mobile and wireless [4]. There exist several architectures for broadcast systems differing in the data delivery methods and mechanisms, such as the push or pull data transmission model, periodic or aperiodic transmission, etc. Each architecture provides advantages for specific environments and/or set of services.

In this work, we consider a popular and powerful broadcast system, suitable for satellite systems: an asymmetric, push-based system with receive-only i clients (i.e. without uplink bandwidth), who are mobile, or in general, connected occasionally. Typical broadcast systems in this category include the deployed systems Pointcast [28], Traffic Information Systems [38], Stock, Weather and News dissemination systems, DirecPc by Hughes [16] and Internet-over-Satellite (IoS) by Intracom [29]. In analogy to a web-type environment, we assume that application information is composed of logical objects (e.g., pages, pictures, text, segments, etc.), where each object is characterized by a different "popularity".

An important technical issue in such broadcast systems is scheduling, i.e. the order in which data objects are transmitted. The server transmits objects, so that their mean aggregate reception delay is minimized for all users. This criterion, the minimized mean aggregate reception delay, is important not only for performance but for energy consumption as well: minimized delay implies that client devices are switched on for a minimized time, and thus, energy consumption of users is minimized.

In push-based broadcast systems, servers broadcast objects in periodic cycles consisting of T time units. Several algorithms have been used to construct the exact transmission schedule in a cycle, given as optimization criterion the minimization of the mean aggregate delay to start receiving an object, also called the access time (the results presented in the following, also apply to non-cyclic schedules, effectively when $T \to \infty$). The schedule of a cycle (period) includes multiple transmissions of each object, depending on its popularity [42]. It is known that, the optimal mean access time is achieved when all appearances of an object are equally distanced within the cycle [30].

Existing analyses of broadcast systems consider *memory-less* clients, i.e. clients which are not equipped with any storage. However, technological advances have led to the deployment of clients with memory today, and actually, memory sizes in the client devices are continuously increasing; in the following, we call this memory a *cache*, because it is not used for long-term storage. Existence of a cache at a client changes the model of a broadcast system, because a caching client is able to start collecting an object even if he/she turns on their device during the transmission of the desired object. This can be achieved provided that, each object transmission is done using packets that have headers with the appropriate packet information. This provision is the state of the art though, because often, transmission is done with fixed size cells, also called radio units, where each cell has an appropriate header, e.g. in GPRS [23], 802.11, etc.

In this chapter subsection, we present three main contributions. First, we provide a simple proof for the need of periodicity (equal distance in transmission) of popular objects in a cycle; the existent proof [30] uses arguments which are very complex. Second, in contrast to existing results, we consider the scheduling problem for caching clients. As we show, the existence of cache not only reduces the reception time of an object, but leads to an optimal broadcast schedule that is different from the optimal schedule for cache-less (traditional) clients. In our work, we calculate the reduced reception delay and we derive the i property that the optimal schedule for caching clients is based on, which is different from the one for cache-less clients. We also describe the scheduler that achieves the optimal schedule. For our work, we use as optimization parameter the mean aggregate reception delay, i.e. the sum of the access delay and the actual object reception time. In prior analyses [42],[46], where models similar to teletext were used, caching clients were not considered, because it was assumed that access time is significantly larger than actual object reception delay; however, in modern systems, this assumption does not hold because it is often necessary to transmit large objects with high popularity. Importantly, our optimization parameter reflects power consumption more realistically than existing models and calculations. This occurs because objects have variable sizes and thus, reception delay is not equal for all objects; however, prior work does not include this extra delay (and corresponding power consumption), because they consider environments with cache-less clients, where the only variable time is the access time.

Our third contribution refers to the analysis of pre-emptive scheduling, among other heuristics, for broadcast systems with or without caching clients. Since perfect periodicity of the broadcasting of all objects within a cycle is an NP-hard problem [9], we prove that the mean aggregate tuning delay of an object in a broadcast schedule may be further reduced, if we allow interruption (pre-emption) of an object's transmission in order to transmit on schedule another more popular one. This is contrary to the usual practice to transmit objects non-preemptively (without interruption). We deduce the conditions under which pre-emption is advantageous and we show that switching the transmission order of two consecutive objects is beneficial in some cases. Finally, we prove that interleaving transmission of two consecutive objects is not advantageous in any case; such interleaving is tempting, considering object packetization and the popularity of interleaving in ATM networks.

The paper is organized as follows. Subsection 3.2 presents an overview of data delivery architectures for broadcast systems and introduces the scheduling problem. Subsection 3.3 introduces the model of the broadcast system, which we analyze, and the notation used in the paper. Subsection 3.4 presents our simple proof of perfect periodicity. Subsection 3.5 presents our analysis for caching clients, including the calculation of aggregate reception delay and the scheduler for achieving optimal schedules in the new model. Finally, Subsection 3.6 describes the conditions under which pre-emptive scheduling provides improved results.

3.2 Data Delivery Architectures

Broadcast systems can be classified, in general, using 3 parameters:

- the data request model (push vs. pull);
- the timing properties of their scheduling scheme (periodic vs. aperiodic);
- the connection model between server(s) and client(s) (unicast vs. 1-to-N).

Every broadcast system is characterized by a specific choice for each of these parameters, leading to 8 possible system configurations. In the following, we elaborate on these three parameters.

A broadcast system is characterized by its data request model, where either the client requests (pulls) data from the server by posting a request, or the server sends (pushes) information to client(s) without explicit requests from the clients. In the pull model, the client posts an explicit request to the server, which, in turn, responds to the client with the requested data; i.e. the client initiates the data transfer. In contrast, in a push system, servers transmit data to clients without a prior request, using some schedule, i.e. the data transfer is initiated by the server itself. The push system is more appropriate for satellite broadcasting to mobile clients, because it eliminates the need for clients to transmit, which is power consuming, especially for the case of GEO satellites.

Data delivery can be periodic or aperiodic in a broadcast system. In periodic systems, data are transmitted according to a predefined schedule (off-line algorithm, similarly to the classic TDMA systems), while in aperiodic systems, data delivery is event-driven, i.e. a data transfer is performed when an event occurs (on-line algorithm, e.g., a request in a pull system or a server decision in a push system). Periodic systems with very long periods can be considered as aperiodic, although they are constructed using an off-line algorithm.

Furthermore, broadcast systems may use different connection models: data transfers may occur in a unicast fashion or in a multicast (broadcast) fashion. In unicast systems, data transfers between servers and clients occur over a one-to-one connection, where no other client can receive the transmitted data. In contrast, in 1-to-N systems, the delivered data are transferred to a set of N clients, which constitute a group. If N is the complete population, then the system follows a broadcast delivery method; however, the broadcast method is typically used in environments where N is unknown.

Several known systems can be classified using the above scheme. Pull-based systems are used, in general, for on-demand services, e.g. Video-on-Demand (VoD), news-on-demand, etc., where clients make requests for a specific service. Actually, these services are typically pull-based and aperiodic, since requests typically arrive at random time instances. Depending on the specific network configuration, the system can be based either on unicast or multicast (1-to-N) connections (a multicast connection is considered as broadcast, when N is the total population). Push-based systems include the characteristic examples of news services and newsgroup services as well as some forms of VoD, where clients have subscribed to receive specific information. Such systems can be either periodic

(sending specific information at designated time intervals) or aperiodic (sending update information only, or information at random intervals).

We focus on push-based, periodic, 1-to-N broadcast systems. These systems are popular in environments where N is unknown. We focus on periodic systems (off-line scheduling algorithms) but our results also apply to effectively non-periodic systems with very large periods. We focus on push-based, because we consider environments where data transmission from clients to servers is expensive and power hungry, e.g. satellites, and client-to-server communication occurs off-line, probably using a less expensive medium, such as a telephone connection. Scheduling in broadcast systems has received a lot of attention.

In pull-based systems, scheduling is necessary to specify the transmission sequence of objects (to choose the object to transmit next). Several scheduling algorithms have been developed for servers [5]: FCFS, Most Requested First (MRF), Most Requested First Lowest (MRFL), Longest Wait First (LWF) and RxW.

Several scheduling algorithms have been developed for push-based systems as well [4, 42, 41]. The criterion is to minimize average access time or tuning time.

One method to minimize access or tuning time, and thus power consumption, is to use indexing methods. In such environments, all objects are identified with a unique key and the system transmits indexing information along with the data; the index is a sequence of pairs of the form (key, location), where the key identifies an object and the location identifies the position of the object in a broadcast cycle. In this fashion, a client can tune in, find the appropriate index entry for the object required and then, it can tune in at the appropriate time, in order to receive the desired object. Several indexing methods have been developed, which differ according to the indexing method or the amount and transmission method of the indexing information. Methods such as (1,m)-Indexing, Distributed Indexing and Flexible Indexing [27] transmit index information in various ways (multiple transmissions of the complete index, distributed transmission of index portions, etc.), while hash-based techniques substitute the indexing information with hashing information that allows the client to calculate the appropriate position of an object in the broadcast cycle.

3.3 Model

We assume a **server** S that broadcasts information **objects** to a set of clients (users), who have subscribed for reception of specific objects off-line. We consider a Web-like environment, where the server transmits objects from a set $O = \{O_1, O_2, \ldots, O_N\}$ and each object O_i is characterized by a **popularity** p_i, which is a probability that quantifies the number of clients waiting to receive a specific object. Transmission is performed using fixed size cells, also called radio units. Objects have arbitrary length and l_i denotes the length of O_i, measured in radio units.

Data transmission is performed in periodic cycles T. In every cycle T, all objects are transmitted and each object may appear more than once in T, de-

pending on its popularity and its length. An appearance of an object in the broadcast cycle is denoted as an **instance** of the object. The **spacing** s_{ij} between two consecutive instances of an object O_i is the time between the beginning of the j-th instance and the beginning of the $(j+1)$-th instance.

Clients in the model are devices which are switched on arbitrarily in time. When a client is switched on, it remains on until it receives the desired object(s), and then it is turned off. The *switch-on* activity of a client can be modeled as posting a request for the object, which the client is waiting for; so, in the following, we will refer to the client switch-on as a *request*. We assume that, during a broadcast cycle T, client requests for an object O_i appear uniformly distributed.

We consider two different types of clients in the system: *caching* clients (equipped with cache) and *cache-less* ones. Cache-less clients need to receive an object from beginning to end, in order to consume it, because they do not have any storage capability. In contrast, caching clients have storage capability and can store partial object information; thus, they are able to store the last part of an object first and then wait for the reception of its beginning later.

We define as **tuning time** for an object, the **access time** of the object (the time until the beginning of reception of the object) plus the actual **reception time** of the object. Thus, we define *waiting time* differently from others, who consider the waiting time equal to the access time. This definition does not affect the construction of the optimal broadcasting schedule for cache-less clients, which are considered in prior work. In our work, we use as optimization parameter the *mean aggregate tuning time*, which is defined as the average of the mean tuning times of all users, i.e. $\sum p_i D_i$, where $i = 1, \ldots, M$, p_i is the *popularity* of object O_i (i.e., the fraction of the user population waiting for O_i at a given time) and D_i is the mean tuning time.

3.4 Periodic Object Transmission in a Cycle

Using the system model with cache-less clients, it has been shown that, for optimal broadcast scheduling (i.e., for minimization of the mean aggregate access delay), all instances of an object should be equally spaced within a cycle T [30].

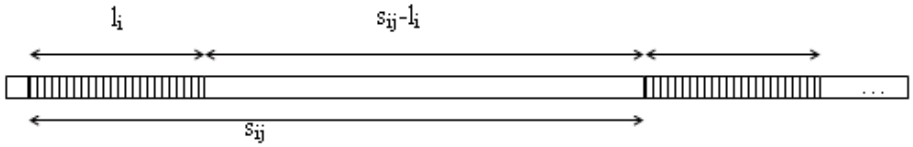


Fig. 22. Calculating the mean tuning delay of object O_i

In the following lemma, we provide a simple proof of this requirement, in contrast to [30], where the proof is quite obscure. Furthermore, we provide a proof

that the same requirement exists for optimal scheduling with caching clients who can start collecting radio units of the desired item as soon as they appear (they start storing parts of the desired object O_i, even if they are switched on during transmission of O_i).

Lemma 1. *The spacing s_i between any two consecutive instances of the same object O_i should be equal in a transmission cycle T.*

Proof. We consider two different cases, depending on whether clients have caches or not.

Cache-less clients

Assume that object O_i is been broadcasted f_i times in a broadcast cycle T. The instances of O_i are at spacings $s_{i1}, s_{i2}, \ldots, s_{if_i}$, where $\sum s_i = T$. If there is **one** request for O_i during T then there is a probability that it will appear during the spacing s_{i1}; this probability is s_{i1}/T, and the mean delay to receive all of O_i is $[l_i + (s_{i1}/2)]$. The same holds true for every spacing s_{ij}. Therefore, the average delay D_i to receive object O_i during a broadcast cycle T is:

$$\begin{aligned} D_i &= (1/T)\{s_{i1}[l_i + (s_{i1}/2)] + s_{i2}[l_i + (s_{i2}/2)] + \cdots + s_{if_i}[l_i + (s_{if_i}/2)]\} = \\ &= (1/T)\{s_{i1}^2/2 + s_{i2}^2/2 + \cdots + s_{if_i}^2/2 + l_i(s_{i1} + s_{i2} + \cdots + s_{if_i})\} = \\ &= (1/T)\{s_{i1}^2/2 + s_{i2}^2/2 + \cdots + s_{if_i}^2/2 + l_i T\} \end{aligned}$$

It is well known that the sum of squares of numbers with a constant sum is minimized when the numbers are equal. Therefore: $D_i = \min$ for $s_{i1} = s_{i2} = \cdots = s_{if_i} = T/f_i$, and so:

$$D_{i\min} = (f_i/2T)(T/f_i)^2 + l_i = (s_i/2) + l_i$$

Caching clients

Under the same assumptions as above (for cache-less clients), the average delay to receive object O_i in a broadcast cycle is:

$$\begin{aligned} D\text{cache}_i = (1/T)\Big\{ & l_i s_{i1} + (s_{i1} - l_i)[(s_{i1} - l_i)/2 + l_i] \\ & + l_i s_{i2} + (s_{i2} - l_i)[(s_{i2} - l_i)/2 + l_i] \\ & \cdots \\ & + l_i s_{if_i} + (s_{if_i} - l_i)[(s_{if_i} - l_i)/2 + l_i] \\ = (1/T)\Big\{ & (l_i \sum s_{ij}) + (s_{i1}^2/2) + (s_{i2}^2/2) + \cdots + (s_{if_i}^2/2) \\ & - l_i(\sum s_{ij}) + (l_i^2/2) f_i + l_i(\sum s_{ij}) - l_i^2 f_i \Big\} \\ = (1/T)\Big\{ & s_{i1}^2/2 + s_{i2}^2/2 + \cdots + s_{if_i}^2/2 + l_i(\sum s_{ij}) - (l_i^2/2) f_i \Big\} \\ = (1/T)\Big\{ & s_{i1}^2/2 + s_{i2}^2/2 + \cdots + s_{if_i}^2/2 + l_i T - (l_i^2/2) f_i \Big\} \end{aligned}$$

This expression is minimized when $s_{i1} = s_{i2} = \cdots = s_{if_i} = s_i = T/f_i$, and so:

$$\begin{aligned} D\text{cache}_{i\min} &= f_i (T/f_i)^2/2T + l_i - (l_i^2/2)(f_i/T) = (s_i/2) + l_i - (l_i^2 f_i/2T) \\ &= D_{i\min} - (l_i^2 f_i/2T) = D_{i\min} - (l_i^2/2s_i) \end{aligned}$$

The last expression shows that, local caching clients receive the desired object O_i with reduced (shorter) delay. The scheduling algorithm presented in [42], which requires that $s_i^2 p_i / l_i = \text{constant}$, does not take into account local memory (cache). If one wants to take advantage of a client's cache, one must modify the condition, on which the scheduling algorithm is based, accordingly. As we prove in the following subsection, the optimal broadcast schedule for caching clients requires that $s_i^2 p_i / l_i + l_i p_i = \text{constant}$. The difference between the two conditions becomes significant, if there exist lengthy objects with high popularities in the system.

3.5 Broadcast System with Caching Clients

Theorem 1. *The optimum broadcast schedule in a system with caching clients requires that*

$$s_i^2 p_i / l_i + l_i p_i = \text{constant}$$

Proof. Our proof follows the lines of the proof for cache-less clients [42] Assume that the following holds for all objects scheduled in a cycle T: the spacing s_i between all pairs of consecutive transmissions of the same object O_i within T is equal (as we will see this is not always possible, but this is approximately true for large broadcast cycles). Then, the average delay to receive object O_i is:

$$D\text{cache}_i = \big(s_i/2\big) + l_i - \big(l_i^2 f_i / 2T\big)$$

So, the mean aggregate delay for all items is:

$$\begin{aligned} D\text{cache} &= \sum D\text{cache}_i p_i = \\ &= \sum \big(s_i p_i/2\big) + \sum l_i p_i - \sum \big(l_i^2 f_i p_i / 2T\big) = \\ &= \sum l_i p_i + \big(1/2\big) \sum p_i \big[s_i - (l_i^2 f_i / T)\big] = \\ &= \sum l_i p_i + \big(1/2\big) \sum p_i l_i \big[(s_i/l_i) - (l_i f_i / t)\big] \end{aligned}$$

We denote as q_i the quantity $q_i = l_i f_i / T$. Clearly: $\sum q_i = \sum (l_i f_i / T) = T^{-1} \sum l_i f_i = 1$. Then, $s_i / l_i = s_i f_i / l_i f_i = T / l_i f_i = q_i^{-1}$. This results to:

$$D\text{cache} = \sum p_i l_i + (1/2) \sum p_i l_i [q_i^{-1} - q_i]$$

In order to find the q_i which minimize the mean aggregate delay Dcache, we set

$$(\partial D\text{cache} / \partial q_i = 0, \text{for } i = 1, 2, \ldots, M$$

So, we find that the following relation must be true:

$$p_1 l_1 (1 + q_1^{-2}) = p_2 l_2 (1 + q_2^{-2}) = \cdots = p_M l_M (1 + q_M^{-2}) = \text{constant}$$

This leads to:

$$\begin{aligned} p_i l_i (1 + q_i^{-2}) = p_i l_i + p_i l_i (T^2 / l_i^2 f_i^2) = p_i l_i + p_i l_i (s_i^2 f_i^2 / l_i^2 f_i^2) = \\ = p_i l_i + p_i s_i^2 / l_i = \text{constant} \end{aligned}$$

Considering the condition of the theorem, one can easily create an on-line scheduling algorithm that constructs optimal schedule for caching clients. We follow the method and the notation of [41]:
let Q denote the current time; the algorithm below decides which object to transmit at time Q. Let $R(j)$ denote the time at which an instance of object O_j was most recently transmitted and $H(j)$ denote the quantity $H(j) = \{[Q - R(j)]^2 p_j/l_j\} + p_j l_j, j = 1, 2, \ldots, M$ ($R(j) = -1$ if Q_j was never transmitted before).
On-line scheduling algorithm:
Step 1: Calculate $H_{\max} = \max\{H(j)\}$, for all j
Step 2: Choose O_i such that $H(i) = H_{\max}$ (if this equality holds true for more than one object, then choose one of them arbitrarily)
Step 3: Transmit object O_i at time Q
Step 4: $R(i) = Q$, go to Step 1

Observe that $[Q - R(j)]$ is the spacing between the current time and the time at which O_i was previously transmitted. The algorithm tries to keep constant the quantity

$$H(j) = \{[Q - R(j)]^2 p_j/l_j\} + p_j l_j$$

This quantity is similar to $p_i s_i^2/l_i + p_i l_i$, which must be constant according to the theorem.

3.6 Pre-emptive Object Scheduling

The optimum schedule requires that the consecutive instances of an object O_i are equally spaced within a broadcast cycle T. This is a very desirable property, especially for energy-limited users, because it reduces *busy-waiting* (as opposed to *stand-by* or *doze-waiting*) for the desired object. However, the design of an optimum schedule is an NP-hard problem [9], leading to use of heuristics that do not achieve the "perfect" schedule. So, as illustrated in Figure 23, it is very probable that an instance of an object (O_2 with length l_2 in the figure) will be broadcasted after the expected time (for perfect periodicity) with high probability, because the transmission of another object (O_1 in the figure) is in progress and must be completed first.

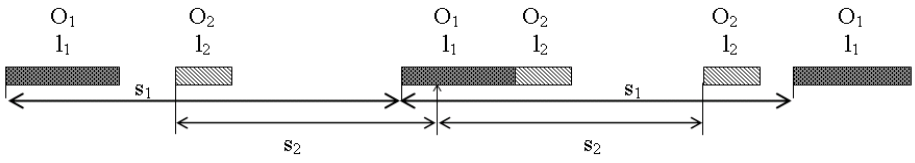


Fig. 23. A case where perfect scheduling is impossible

One can attempt to improve the schedule with the following heuristics:

1. Interrupt the transmission of object O_1, transmit part of O_2, complete the transmission of O_1 and then of O_2 (see Figure 24). As we prove below, this *interleaving* is not the right approach, because it always increases the total delay. It is easy to see that the same conclusion holds true if we attempt to do a finer interleaving.
2. Interrupt the transmission of O_1, transmit the complete O_2, and then resume and complete the transmission of O_1 (see Figure 25). This is an example of *pre-emptive transmission*. As we prove below, this approach may decrease the total delay under the right circumstances.
3. Transmit O_2 first (ahead of schedule) and then O_1 (behind schedule) (see Figure 3.6). Again, this ahead-of-schedule transmission decreases the total delay under certain conditions.

The results mentioned for each heuristic above hold for both client models, caching and cache-less. In the remaining subsection, we analyze all heuristics for the case of cache-less clients, because this analysis is much easier to present. We also present, for comparison, as case 4, the analysis of pre-emptive transmission for caching clients and derive the condition under which the total delay is reduced. In all cases analyzed below, only the delays of clients waiting for objects O_1 and O_2 are influenced. Remaining clients do not experience any difference in performance.

Case 1: Interleaving (Cache-less clients)

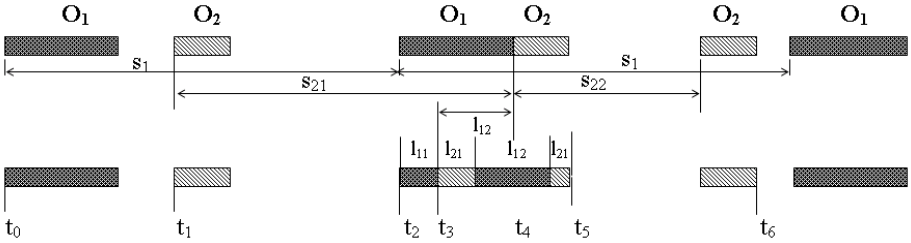


Fig. 24. Interleaving objects O_1 and O_2

Calculation of the increase of the average delay for object O_1:
Requests for object O_1 appearing in the interval $[t_0, t_2]$ will experience a delay **increase** to receive the object; this increase is equal to l_{21}. Remaining requests are unaffected. Therefore, the increase of the mean aggregate delay is equal to: $p_1 s_1 l_{21}/T$.

Calculation of the increase of the average delay for object O_2:
Requests for O_2 that appear in the interval $[t_3, t_4]$ will experience a delay **increase** to receive the object equal to: $t_6 - t_5 = s_{22}$ (s_{22} is the spacing between instances #2 and #3). The remaining requests are unaffected. Given that the interval $[t_3, t_4]$ has length l_{12}, the increase of the mean aggregate delay is equal to: $p_2 s_{22} l_{12}/T$.

Combining the results of the two previous calculations, we see that it is not appropriate to interrupt the transmission of O_1 in order to transmit part of O_2, because this decision always increases the mean aggregate delay by the amount:

$$\text{Total delay increase} = [p_1 s_1 l_{21} + p_2 s_{22} l_{12}]/T > 0$$

Case 2: Pre-emption (Cache-less Clients)
In this case, the transmission of O_1 is interrupted, O_2 is transmitted and then the transmission of O_1 is completed, as depicted in Figure 25.

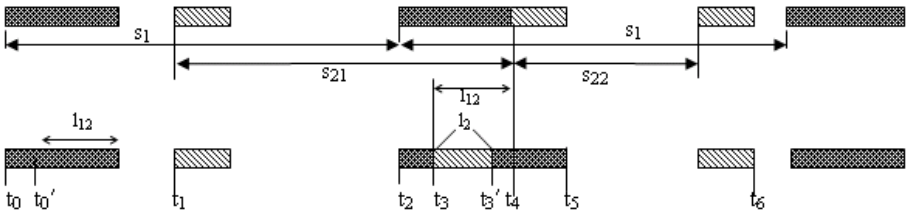


Fig. 25. Pre-emptive transmission of O_2

Calculation of the increase of the average delay for object O_1:
Requests for O_1 appearing in the interval $[t_0, t_2]$ will experience a delay **increase** equal to l_2, in order to receive O_1. The remaining requests are unaffected. Therefore, the increase of the mean aggregate delay for O_1 is equal to: $p_1 s_1 l_2/T$.

Calculation of the increase of the average delay for object O_2:
Requests for O_2 appearing in the interval $[t_1, t_3]$ (with length $s_{21} - l_{12}$) will experience a delay **decrease** equal to l_{12}, in order to receive O_2. Requests made during $[t_3, t_4]$ will experience a delay **increase** equal to s_{22} to receive O_2. The remaining requests are unaffected. Therefore, the **increase** of the mean aggregate delay is:

$$[-p_2(s_{21} - l_{12})l_{12} + p_2 l_{12} s_{22}]/T$$

Given the above calculations, the total **increase** of the mean aggregate delay ΔD is:

$$\begin{aligned} \Delta D &= [p_1 s_1 l_2 - p_2(s_{21} - l_{12})l_{12} + p_2 l_{12} s_{22}]/T = \\ &= [p_1 s_1 l_2 + p_2 l_{12}^2 - p_2(s_{21} - s_{22})l_{12}]/T \end{aligned}$$

So, the delay increase depends on l_{12}. If we set $d\Delta D/dl_{12} = 0$, then we find that ΔD takes a minimum value, if $l_{12} = (s_{21} - s_{22})/2$. This minimum value is:

$$\Delta D_{\min} = [p_1 s_1 l_2 - p_2 (s_{21} - s_{22})^2]/4T$$

and is negative (= **maximum decrease of delay**) if

$$(s_{21} - s_{22})^2 > 4 p_1 s_1 l_2 / p_2$$

Thus, if this inequality holds, then we can reduce the mean aggregate delay, if we interrupt the transmission of O_1 after $l_1 - l_{12} = l_1 - (s_{21} - s_{22})/2$ radio units, in order to transmit O_2.

Case 3: Ahead-of-Schedule Transmission (Cache-less Clients)
In this case, we transmit O_2 ahead of schedule and O_1 behind schedule.

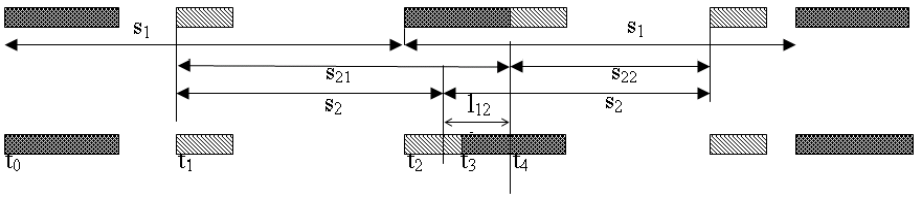


Fig. 26. Switching the transmission order of O_1 and O_2

Calculation of the increase of the average delay for object O_1:
Requests for O_1 appearing in the interval $[t_0, t_2]$ will experience a delay **increase** equal to l_2, in order to receive O_1. Requests in the interval $[t_2, t_3]$ will experience a delay **decrease** equal to $(s_1 - l_2)$, while all remaining requests are unaffected. Therefore, the increase of the mean aggregate delay is equal to:

$$[p_1 s_1 l_2 - p_1 l_2 (s_1 - l_2)]/T$$

Calculation of the increase of the average delay for object O_2:
Requests for O_2 appearing in the interval $[t_1, t_2]$ will experience a delay **decrease** equal to l_1 to receive O_2. Requests in the interval $[t_2, t_4]$ will experience a delay **increase** equal to s_{22}. All remaining requests are unaffected. Therefore, the increase of the mean aggregate delay is equal to:

$$[-p_2 l_1 (s_{21} - l_1) + p_2 l_1 s_{22}]/T$$

The total delay increase is:

$$\begin{aligned} \Delta D &= [p_1 s_1 l_2 - p_1 l_2 (s_1 - l_2) - p_2 l_1 (s_{21} - l_1) + p_2 l_1 s_{22}]/T = \\ &= [p_1 l_2^2 + p_2 l_1^2 - p_2 l_1 (s_{21} - s_{22})]/T \end{aligned}$$

This expression becomes negative (**decrease of the delay**) if

$$(s_{21} - s_{22}) > l_1 + (p_1 l_2^2)/p_2 l_1$$

Since $(s_{21} - s_{22}) = 2l_{12}$ (see Figure 3.6), we obtain:

$$l_{12} > [l_1 + (p_1 l_2^2)/p_2 l_1]/2$$

If this inequality holds, the mean aggregate delay is **reduced** when the transmission order of O_1 and O_2 is switched.

The results of this analysis hold true for the case of caching clients as well; however, the analysis is more involved. We present the analysis of pre-emptive transmission with caching clients.

Case 4: Pre-emption (Caching Clients)
Consider the pre-emptive transmission of object O_2, shown in Figure 25, for the case of caching clients. We derive the conditions under which this pre-emptive transmission reduces the total tuning time.

Calculation of the increase of the average delay for object O_1:
Clients requesting O_1 and appearing in the interval $[t'_0, t_2]$ will experience an **increase** of the tuning time by l_2. Requests during the interval $[t_2, t_3]$ will not be affected. Requests during $[t_3, t'_3]$ will have a mean **decrease** of their tuning time by $l_2/2$. Similarly, requests during $[t'_3, t_4]$ will have a **decrease** of their tuning time by l_2, while requests during $[t_4, t_5]$ will have a mean decrease by $l_2/2$. Putting these together, we see that the mean increase of the tuning time of clients requesting O_1, due to the pre-emptive transmission of O_2 is:

$$\Delta D_1 = (p_1/T) l_2[(s_1 - l_1 + l_{12}) - l_2/2 - (l_{12} - l_2) - l_2/2] = p_1 l_2 (s_1 - l_1)/T$$

Calculation of the increase of the average delay for object O_2:
Clients requesting O_2 and arriving in the interval $[t_1, t_3]$ will have a **decrease** of their tuning time by l_{12}. Requests during the interval $[t_3, t'_3]$ will have a mean **increase** of their tuning time by $(s_{22} - l_2/2)$, requests during the interval $[t'_3, t_4]$ an **increase** by s_{22}, and requests during the interval $[t_4, t_5]$ a mean **increase** by $l_2/2$. Thus, the mean **increase** of tuning time for object O_2 is:

$$\begin{aligned}\Delta D_2 &= (p_2/T)[-(s_{21} - l_{12}) l_{12} + (s_{22} - l_2/2) l_2 + (l_{12} - l_2) s_{22} + l_{22}/2] = \\ &= p_2 l_{12} (s_{22} - s_{21} + l_{12})/T\end{aligned}$$

Working similarly to the case of cache-less clients, we find that the condition under which pre-emptive transmission **reduces** the total tuning time is:

$$(s_{21} - s_{22})^2 > 4 p_1 (s_1 - l_1) l_2 / p_2$$

Thus, the maximum reduction of the tuning time is achieved when: $l_{12} = (s_{21} - s22)/2$.

3.7 Conclusions

We showed that, as expected, a local cache reduces the time required for the reception of an object. We also proved that the optimal broadcast schedule for these caching clients is different from the optimal schedule for cache-less clients (it must satisfy the condition $s_i^2 p_i/l_i + p_i l_i = \text{constant}$, rather than the condition $s_i^2 p_i/l_i = \text{constant}$). Considering a broadcast schedule constructed with heuristic algorithms that are based on these conditions, we proved that, for caching or cache-less clients, the mean aggregate delay for the complete reception of an object may be further reduced, if we allow the interruption of the transmission of an object in order to transmit on schedule another more popular one. We deduced the conditions under which this pre-emption reduces the mean aggregate delay. Furthermore, we showed that under some conditions the switching in the transmission order of two neighboring objects may also be advantageous. Finally, we proved that the interleaving of the transmissions of two neighboring objects is not beneficial in any case.

4 Acknowledgments

The authors would like to thank professor Jon Crowcroft for his detailed comments.

References

[1] Freeflow: How it works. Akamai, Cambridge, MA, USA, November 1999.

[2] C. C. Aggarwal, J. L. Wolf, and P. S. Yu. A permutation-based pyramid broadcasting scheme for video-on-demand systems. In *proceedings of ICMCS*, pages 118–126, June 1996.

[3] Charu C. Aggarwal, Joel L. Wolf, and Philip S. Yu. On optimal batching policies for video-on-demand storage servers. In *Proceedings of ICMCS*, pages 253–258, Hiroshima, Japan, June 1996.

[4] D. Aksoy, M. Altinel, R. Bose, U. Cetintemel, M.J. Franklin, J. Wang, and S.B. Zdonik. Research in Data Broadcast and Dissemination. In *Proceedings of the First International Conference on Advanced Multimedia Content Processing (AMCP'98)*, Lecture Notes in Computer Science, pages 194–207. Springer Verlag, 1999.

[5] D. Aksoy and M. Franklin. Scheduling for Large-Scale On-Demand Data Broadcasting. In *Proceedings of INFOCOM 1998*, San Francisco, CA, 1998.

[6] K. C. Almeroth and M. H. Ammar. On the use of multicast delivery to provide a scalable and interactive video-on-demand service. *IEEE Journal on Selected Areas in Communications*, 14(6):1110–1122, April 1996.

[7] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proceedings of ACM SIGCOMM*, August 2002.

[8] Sujata Banerjee, Jack Brassil, Amy C. Dalal, Sung Ju Lee, Ed Perry, Puneet Sharma, and Andrew Thomas. Rich media from the masses. Technical Report HPL-2002-63R1, HP Lab, May 2002.

[9] A. Bar-Noy, B. Randeep, J. Naor, and B. Schieber. Minimizing service and operation cost of periodic scheduling. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 11–20, 1998.

[10] Ernst Biersack, Alain Jean-Marie, and Philippe Nain. Open-loop video distribution with support of vcr functionnality. *Performance Evaluation*, 49(1-4):411–428, September 2002.

[11] Yitzhak Birk and Ron Mondri. Tailored transmissions for efficient near-video-on-demand service. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, pages 226–231, june 1999.

[12] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron, and Atul Singh. Splitstream: High-bandwidth content distribution in a cooperative environment. In *proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS)*, March 2003.

[13] G. Chan and F. Tobagi. Distributed servers architectures for networks video services. *IEEE/ACM Transactions on Networking*, 9(2):125–136, April 2001.

[14] Yang H. Chu, Sanjay G. Rao, Srinivasan Seshan, and Hui Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *proceedings of ACM SIGCOMM*, San Diago, CA, August 2001.

[15] Dash Optimization. *Xpress-Mp Essentials*, 2001.

[16] DirecPC. URL: `http://www.direcpc.com/index.html`.

[17] D. Eager, M. Vernon, and J. Zahorjan. Optimal and efficient merging schedules for video-on-demand servers. In *proceedings of the 7th ACM Multimedia Conference*, November 1999.

[18] Derek Eager, Mary Vernon, and John Zahorjan. Minimizing bandwidth requirements for On-Demand data delivery. *IEEE Transactions on Knowledge and Data Engineering*, 2001.

[19] L. Gao and D. Towsley. Threshold-based multicast for continuous media delivery. *IEEE Transactions on Multimedia*, 3(4):405–414, December 2001.

[20] V.K. Goyal. Multiple description coding: compression meets the network. *IEEE Signal Processing Magazine*, 18(5):74 –93, September 2001.

[21] Yan Guo, Subhabrata Sen, and Don Towsley. Prefix caching assisted periodic broadcast: Framework and techniques for streaming popular videos. In *proceedings of IEEE ICC*, April 2002.

[22] Yang Guo, Kyoungwon Suh, Jim Kurose, and Don Towsley. P2cast: Peer-to-peer patching scheme for vod service. In *Proceedings of the 12th World Wide Web Conference (WWW)*, Budapest, Hungary, May 2003.

[23] G. Heine. *GPRS from A to Z*. Artech House Inc., April 2000.

[24] Ailan Hu. Video-on-Demand broadcasting protocols: A comprehensive study. In *proceedings of Infocom*, volume 1, pages 508–517, Anchorage, Alaska, USA, April 2001.

[25] K. A. Hua and S. Sheu. Skyscraper broadcasting for metropolitan vod. In *proceedings of Sigcomm*, August 1997.

[26] Kien A. Hua, Ying Cai, and Simon Sheu. Patching : A multicast technique for true video-on-demand services. In *proceedings of ACM Multimedia*, pages 191–200, 1998.

[27] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Energy Efficient Indexing on Air. *ACM SIGMOD Record*, 23(2):25–36, June 1994.

[28] Infogate. URL: `http://www.pointcast.com/`.

[29] Intracom S.A. URL: `http://www.intranet.gr/en/products/internet/ios.htm`.

[30] R. Jain and J. Werth. Airdisks and airRAID: Modeling and scheduling periodic wireless data broadcast. Technical report, DIMACS Technical Report 95-11, May 1995.

[31] John Jannotti, David K. Gifford, and Kirk L. Johnson. Overcast: Reliable multicasting with an overlay network. In *proceedings of the 4-th Symp. on Operating Systems Design and Implementation*. Usenix, Octobre 2000.

[32] L. Juhn and L. Tseng. Fast data broadcasting and receiving scheme for popular video service. *IEEE Trans.on Broadcasting*, 44(1):100–105, March 1998.

[33] Li-Shen Juhn and Li-Ming Tseng. Harmonic broadcasting for video-on-demand service. *IEEE Transactions on Broadcasting*, 43(3), September 1997.

[34] K. Kong and D. Ghosal. Mitigating server-side congestion in the internet through pseudoserving. *IEEE/ACM Transactions on Networking*, pages 530–544, August 1999.

[35] J. Nussbaumer, B. Patel, F. Schaffa, and J. Sterbenz. Networking requirements for interactive video on demand. *IEEE Journal on Selected Areas in Communications*, 13(5):779–787, 1995.

[36] V. N. Padmanabhan, Helen J. Wang, Philip A. Chou, and Kunwadee Sripanidkulchai. Distributing streaming media content using cooperative networking. In *proceedings of NOSSDAV*, May 2002.

[37] Sridhar Ramesh, Injong Rhee, and Katherine Guo. Multicast with cache (mcache): An adaptive zero delay video-on-demand service. *IEEE Transactions of Circuit and System of Video Transmission*, 11(3), March 2001.

[38] S. Shekhar and D. Liu. Genesis and Advanced Traveler Information Systems ATIS: Killer Applications for Mobile Computing. MOBIDATA Workshop, 1994.

[39] Simon Sheu and Kien A. Hua. Virtual batching: A new scheduling technique for video-on-demand servers. In *Database Systems for Advanced Applications*, pages 481–490, April 1997.

[40] TiVo. What is tivo: Technical aspects, 2003.

[41] N.H. Vaidya and S. Hameed. Data broadcast in asymmetric wireless environments. In *Proceedings of Workshop on Satellite-based Information Services (WOSBIS)*, New York, November 1996.

[42] N.H. Vaidya and S. Hameed. Scheduling data broadcast in asymmetric communication environments. *ACM/Baltzer Wireless Networks*, 5(3):171–182, 1999.

[43] S. Viswanathan and T. Imielinski. Pyramid broadcasting for video on demand service. In *Proceedings of Multimedia Conference*, San Jose, CA, February 1995.

[44] Bing Wang, Subhabrata Sen, Micah Adler, and Don Towsley. Proxy-based distribution of streaming video over unicast/multicast conncetions. In *proceedings of Infocom*, june 2002.

[45] Paul P. White and Jon Crowcroft. Optimized batch patching with classes of service. *ACM Communications Review*, October 2000.

[46] J. W. Wong. Broadcast delivery. *Proceedings of the IEEE*, 76:1566–1577, December 1999.

[47] Dongyan Xu, Heung-Keung Chai, Catherine Rosenberg, and Sunil Kulkarni. Analysis of a hybrid architecture for cost-effective streaming media distribution. In *proceedings of SPIE/ACM Conference on Multimedia Computing and Networking (MMCN)*, Santa Clara, CA, January 2003.

[48] Y. Zhao, D. Eager, and M. Vernon. Network bandwidth requirements for scalable on-demand streaming. In *proceedings of Infocom*, june 2002.

Pricing and QoS

Burkhard Stiller (Ed.)[1], Pere Barlet-Ros[2], John Cushnie[3], Jordi Domingo-Pascual[2], David Hutchison[3], Rui Lopes[3], Andreas Mauthe[3], Mihai Popa[4], Jim Roberts[5], Josep Solé-Pareta[2], Denis Trcek[6], Carlos Veciana[2]

[1] Information Systems Laboratory IIS,University of Federal Armed Forces Munich
Werner-Heisenberg-Weg 39, D-85577 Neubiberg, Germany
stiller@informatik.unibw-muenchen.de
http://www.informatik.unibw-muenchen.de, and
Computer Engineering and Networks Laboratory TIK
ETH Zürich, Gloriastrasse 35, CH-8092 Zürich, Switzerland
stiller@tik.ee.ethz.ch, http://www.tik.ee.ethz.ch
[2] Department Arquitectura de Computadors,
Advanced Broadband Communications Laboratory (CCABA)
Universitat Politècnica de Catalunya (UPC)
Jordi Girona, 1-3, Mòdul D6 (Campus Nord), 08034 Barcelona, Catalunya, Spain
{pbarlet,carlosv,pareta,jordid}@ac.upc.es
http://www.ccaba.upc.es
[3] Computing Department, Lancaster University
SECAMS Building, Lancaster, LA1 4YR, UK
{j.cushnie, a.mauthe, r.lopes, d.hutchison}@lancaster.ac.uk
http://www.comp.lancs.ac.uk/computing/
[4] R&D Department, SC PROCETEL SA
Calea Rahovei 266-268, Bucharest, Romania
mihpopa@fx.ro
[5] France Telecom R&D, DAC/OAT
38 rue du General Leclerc, 92794 Issy-Moulineaux, France
james.roberts@francetelecom.com
http://perso.rd.francetelecom.fr/roberts/
[6] Department of Digital Communications and Networks E6
Jozef Stefan Institute, Jamova 39, 1101 Ljubljana, Slovenia
denis.trcek@ijs.si, http://epos.ijs.si/

Abstract. In this chapter the state of the art of pricing for Internet services and its relation to Quality-of-Service (QoS) is addressed. Essential economic and technology basics, covering terms, accounting, and security are followed by a user-centered view, a content-based scheme, and a cost sharing approach.

1 Introduction

To offer different Quality-of-Service (QoS) levels within a network implies different prices to be paid for these levels, while allowing users to choose and control what best meets their QoS requirements and payment possibilities. The QoS actually achieved depends on how much network capacity is provided to meet the expressed demand, on the current traffic load, or on QoS mechanisms provided within a network to guarantee requirements. Therefore, the tasks of charging, pricing, and billing for

M. Smirnov et al. (Eds.): COST 263 Final Report, LNCS 2856, pp. 263-291, 2003.

Internet services, including the mobile Internet, have been identified as an important research area over recent years. The global provision of information and services in a private and commercial fashion is a great motivator for industry and academia alike, but to make the most out of this opportunity, efficient methods for charging and billing need to be proposed, developed, and deployed [1], [2], [6], [34], and [15].

For the commercial Internet use, which is managed by network providers and Internet Service Providers (ISP), users may be charged with a specific tariff. These tariffs may be based on traditional schemes, such as time-based or leased-line charges, or on new concepts related to IP networks, such as volume or other traffic characteristics. Moreover, additional scenarios for applying charging are driven by the content offered by an Internet service on top of the transport network, *e.g.*, including web access, e-commerce, video conferencing. Charging for these services involves accurate end-to-end tracing of the service offered, even more in some cases QoS guarantees, inter-network provider, and inter-ISP charging agreements.

Besides (a) the technical service differentiation perspective, two major economic motivations for Internet charging exist: (b) the recovery of the cost of investment to provide the service and (c) the generation of profit for companies providing these services. Finally, (d) the operational view point includes the need of a provider to provide congestion control in the Internet, which is possible besides traditional technical means through differentiating service levels according to price and congestion pricing. With respect to the relation between technology and economic points of views, pricing and provisioning are obviously related since the former must generate sufficient revenue to cover the cost of the latter. In addition, two particularly different scenarios for providers can be distinguished: the commercial Internet and National Research Network (NRN) providers. Therefore, major influencing factors for pricing differentiated services, QoS-enabled services, or different networks include the following ones:

- **Technical:** QoS guarantees, accounting, and security mechanisms.
- **Economic:** Incentive-compatible pricing schemes, tariff models, and charges.
- **Organizational:** Congestion control mechanisms, customer care, and billing.
- **User-oriented:** Ease-of-use, transparency, and application-independence.

In addition, single or multi-provider differentiation is important to distinguish between inter- and intra-domain activities to take place for providing charged services. The customer (or user) and provider perspective are critical with respect to the transparency and ease-of-use of pricing models and resulting charge calculations. On one hand, a provider is interested to know about QoS, user demands, and prices users are prepared to pay for. On the other hand, providers should offer a range of QoS, associated to different pricing schemes, allowing and asking users to choose what best meets their demands and financial budget. Aspects on user-driven price control and new provider-user relationships will enhance the view of Internet pricing models.

1.1 Terminology

To avoid any misconceptions on terms utilized within this chapter, the following list of definitions is used for key terms in the area of pricing for QoS [39]:

- **Metering** determines the particular usage of resources within end-systems (hosts) or intermediate systems (routers) on a technical level, including Quality-of-Service (QoS), management, and networking parameters.
- **Accounting** defines the summarized information (accounting records) in relation to a customer's service utilization. It is expressed in metered resource consumption, *e.g.*, for the end-system, applications, calls, or any type of connections.
- **Charge Calculation** covers the calculation of a price for a given accounting record and its consolidation into a charging record, while mapping technical values into monetary units. Charge calculation applies a given tariff to the data accounted for.
- **Charging** is an overall term, depicting all tasks required to calculate the finalized content of a bill at a higher layer of abstraction.
- **Pricing** covers the specification and setting of prices for goods, in this chapter specifically networking resources and services in an open market situation. This process may combine technical considerations, *e.g.*, resource consumption, and economic ones, such as applying tariffing theory (prices calculated on a cost/profit base) or marketing.
- **Tariff** defines the algorithm used to determine a charge for a service usage. It is applied in the charge calculation for a given customer and service he utilizes. Tariffs may contain, *e.g.,* discount strategies, rebate schemes, or marketing information.
- **Billing** defines the collection of charging records, summarizing their charging content, and delivering a bill/invoice including an optional list of detailed charges.

2 Economic Basics

To help understand Internet charging and billing issues, the wide range of proposed and existing pricing schemes is discussed. The essential pricing function of return on investment is considered, which is clearly a critical consideration for a commercial operator. Cost sharing is a particularly important issue, when institutions share a common infrastructure, as in the case of an NRN (National Research Network). A brief overview of current projects related to these issues concludes this section.

2.1 Pricing Schemes

While most ISPs currently practice flat rate pricing, at least for residential users, it is widely accepted that the introduction of advanced services with differentiated QoS will lead to the development of more sophisticated schemes [40]. Briefly, the advantages and drawbacks of flat rate pricing are outlined, before discussing respective principles of usage-based pricing, congestion pricing, and service-related pricing. Another overview on pricing may be obtained from [17].

2.1.1 Flat Rate Pricing

Flat rate pricing has the great advantage of simplicity. This facilitates implementation and avoids obvious sources of contention about the bill between customer and

supplier. The bill is perfectly predictable and this brings considerable comfort to users who have no need to keep one eye on the clock or byte counter to avoid an unpleasant surprise at the end of the month. Experiments in moving away from the flat rate scheme to some form of usage-based charging have always been met with considerable consumer resistance.

A principal disadvantage of flat rate is the absence of any direct relation between price and cost. There is no disincentive to prevent users generating an excessive amount of traffic thus requiring increased investment in network infrastructure. Flat rate pricing is unfair when users with vastly different usage, *e.g.*, the peer to peer hacker and the Web surfer, must pay the same price while incurring quite different costs. Finally, flat rate is closely associated with best effort service as there is no mechanism to allow any user who so wishes to pay more to avoid the negative effects of congestion.

2.1.2 Usage-Based Pricing

The main goal of the usage-based pricing model is charging the usage of the network resources. This usage knowledge is acquired by detailed traffic monitoring. One of the most important traffic characteristic to take into account is the traffic volume. Nevertheless, other traffic characteristics can be also considered to obtain a more detailed charging scheme. Some of these characteristics could be the packet origins and destinations, applications, information contents of the packets, etc. One special type of usage-based pricing is the content-based pricing. This model has several difficulties, such as the impossibility of processing encrypted packets, and the legal restrictions. Furthermore, the number of resources needed for content analysis grows drastically in high-speed links.

Usage-based pricing could be desirable in order to make users aware of the implication of their actions on the network. On the other hand, this is also the main argument against usage-based pricing, since users reduce their network usage when they are charged by a usage-based scheme. This could be positive in NRN environments, but the effect is not clear in commercial networks.

2.1.3 Congestion Pricing

Pricing may be used to ensure that a scarce resource is used to produce maximum utility when shared among a population of contending users having different valuations. This is the principle of congestion pricing.

Many Internet pricing schemes have been put forward to achieve such optimal sharing, a notable example being the “smart market” proposal [30]. This is more an economic ideal than a practical scheme, however. More pragmatic congestion pricing principles are included in the DiffServ architecture. [37] suggested that users could express the value of their packets by choosing between a small number of classes with each class being priced depending on its relative quality level. Users would be obliged to choose expensive classes in times of congestion, if their utility justified the cost, but could revert to cheaper classes in slack periods. The notion of expected capacity filter with “in-profile” and “out-of-profile” packets was introduced [10]. Pricing is based on the parameters of the filter, which can be chosen by the user to modulate the proportion of in-profile packets and thus determine realized QoS in times of congestion. An alternative approach to congestion pricing has been proposed recently

[19]. In this scheme, packets are marked when they contribute to a congestion situation and pricing is related to the number of marked packets. Users can modulate their charge by changing their packet rate in response to the current level of congestion.

2.1.4 Service-related Pricing

An alternative pricing principle in a multiservice network is to relate tariffs to required quality levels. Price is related to user-declared traffic characteristics and performance requirements and is logically chosen to reflect the cost of meeting the respective demands. To be credible, such a scheme requires Quality-of-Service differences that are consistent and measurable. Unfortunately, this proves difficult since performance depends in a quite complex way on the statistical characteristics of demand and the amount of resources provisioned to meet that demand. It is relatively easy to ensure excellent Quality-of-Service for all (by over provisioning) but practically impossible to meet precise intermediate levels of performance between excellent and bad. *E.g.*, meeting a packet loss rate exactly between 0.001 and 0.01 without knowing a mathematical relation between demand, capacity, and performance, is not possible.

2.2 Cost Recovery

Installed infrastructure and network operation constitute significant cost items which must be recovered by pricing [41]. A potentially useful model for cost recovery is the charging scheme of the telephone network. Traditional telephone networks are generally uncongested by design so that all revenue comes from non-congestion related pricing, i.e., flat rate or usage-based pricing. Telephone prices are set at a level such that expressed demand (accounting for price elasticity) is within the limits of available capacity and generated revenue is sufficient to cover the cost of that capacity. Time of day price modulation is used to smooth demand somewhat by giving incentives to use the network in off-peak periods. In general, competition drives providers to operate their networks efficiently with just the right degree of over-provisioning to avoid congestion. To follow the example of the telephone network would require ensuring that sufficient capacity is available in the Internet to meet demand, save in exceptional circumstances. Pricing would then logically be essentially usage-based. Alternative mechanisms, such as flow-based admission control, as discussed in Chapter 1, would be necessary to preserve Quality-of-Service in rare cases of overload.

2.3 Cost Sharing

Volume-based charging and billing schemes have been applied mainly in NRN, addressing cost sharing among their universities and research institutions.

The Case of New Zealand: One of the first examples was the case of New Zealand's NRN [7]. This experience has served as a springboard for many other cost-sharing proposals in other regions. Since 1990 the University of Waikato has operated a single Internet gateway to the United States, charging users by volume to recover the costs. This was very successful, allowing the steady growth of this link speed. Each organization predicted the amount of traffic that it would move during one

month. Then a fixed price per unit was paid for that traffic. Also, the real traffic moved was accounted for, and an extra charge per unit was applied for those beyond the predicted traffic. Also, the latest versions of the charging scheme propose different charges for different network services, which can be charged for at a different price.

The JANET Model: JANET, the NRN of the United Kingdom, charged its members by volume, but only for traffic to links in the United States [34], [12]. Charging was applied during high-activity hours of the day in an attempt to reduce traffic and to redistribute traffic to low-activity hours. Traffic classification was based on local Network IP addresses, and every organization charged was able to receive detailed information from some groups of its servers. The system operated until 2000 when it was decided to revise the charging scheme. A combination of fixed and usage-based charges was proposed in order to make the budget process more predictable.

The SWITCH Model: The SWITCH network provides connectivity to universities and research institutions in Switzerland, and it recovers its costs from member institutions [34], [41]. Each institution pays a fixed rate for the connection, only one third of the cost. Moreover, the other two thirds of the cost are charged for based on volume. Therefore, charging is mainly volume-driven. In the case of SWITCH, the effects of volume-based charging have been observed in the behavior of the users.

2.4 Projects Related to Charging

QoS-based charging support, of course in combination with accounting, for premium IP services have also been researched. The SUSIE project [42] resulted in valuable results and observations including user trials and charging schemes appropriate for DiffServ, IntServ, and ATM networks. It supports an accounting and charging platform suitable for audio-visual services in heterogeneous scenarios with wholesale and retail service providers. For the inter-operation issues, SUSIE has outlined a trade accounting architecture. While integrating and validating accountable IP services, a model for convergence charging has been established based on a usage charged ATM network delivering Premium IP with ATM related QoS and charge prioritized ATM streams selected via a QoS/Price trader. The M3I project [6] developed a Charging and Accounting System (CAS) for Internet services, which utilizes metered data originating from network elements of interest. These data are accounted for and depending on a communicated pricing model and tariff the charges are calculated for the service usage. The MobyDick project [32] applies Diameter to account for mobile service data, which are used in turn to calculate charges.

Finally, the special focus on content-based charging reveals that there are several ongoing projects relevant to this research. Various IETF WGs have a focus on content delivery and content charging. The CDI WG aims to define protocols to allow the inter-operation of separately administered content networks, with several Internet Drafts, *e.g.*, [14] or [13], already released but no RFCs to date. The AAA Working Group [2] is driving the Diameter protocol standard for AAA across peering networks, which is also being considered by the CDI WG as the AAA protocol of choice. The WEBI WG is working on defining the Resource Update Protocol (RUP), which may be adopted by the CDI WG for managing the distribution of content

within a CDI network. Within the MPEG standards group [38] the emerging MPEG-7 and MPEG-21 standards complement and contribute to the efforts ongoing with CDI networks. The IST CADENUS project [8] is researching the implementation of SLAs across Premium IP networks, which may also be applied to CDI networks.

3 Technical Basics and Services

The basis for a charging approach is given by a suitable accounting infrastructure, today embedded in existing AAA architecture and protocol proposals, where some of which are available as commercial platforms. In addition, security-relevant services are addressed in combination to ensure that measured data are kept accordingly.

3.1 AAA and Beyond

The IETF and IRTF both have on-going research on an AAA architecture [2] to meet the short- and long-term requirements for the Internet as gathered from the NASREQ (Network Access Server Requirements), MOBILE IP, and ROAMOPS (Roaming Operations) Working Groups. These are currently tending towards Diameter and the earlier RADIUS as the preferred protocols. The need for service requires, in many models, Authentication, to verify a claimed identity, Authorization, to determine if a particular right can be granted to the presenter of a particular credential, and Accounting, to collecting information on resource usage for the purpose of trend analysis, auditing, billing, or cost allocation. Regardless how one function of the three AAA leads to or derives from others, there is common agreement that they are closely interdependent.

Basically, the AAA architecture includes local and home agents and AAA servers that establish secure channels for the purposes of exchanging sensitive (access) information. An agent that attends to the client's request is likely to require that the client provides some credentials that can be authenticated before access to the resources is authorized. Then, accounting information is interchanged. That architecture requires transport-independent AAA protocols meeting requirements on security, scalability, reliability, as well as inter-domain access control. They also have to provide — including a clear documentation — an accounting operations model for each type of network access, the support for IPv6, an explicit proxy support, a data model separated from the protocol, a MIB support, a RADIUS backward compatibility, as well as a full coverage of operational problems by a set of error messages.

Since current AAA architectures, protocols, and implementations do not cope fully with heterogeneous application scenarios and many requirements for various services, ranging from connectivity to content, are not supported, this lack of a generic approach drove the A^x development [35]. It distinguishes between support services and user services and integrates a policy-based management architecture by separating decision points from enforcement points on a per-service basis. So-called A^x services can be offered by a specialized A^x system. A^x services, apart from metering, can be offered from one provider to another because of their future separation based on A^x, enabling providers to build systems driven by their specific business requirements.

3.2 Accounting Platforms

High-speed links that deal with high volumes of traffic belonging to a high number of users from a wide variety of profiles need tools for traffic analysis that can gather traffic information with a high degree of detail. Currently, the most known accounting tools are the Cisco NetFlow technology [9] and the CAIDA's CoralReef Suite [33].

3.2.1 Cisco NetFlow

Cisco IOS NetFlow is a part of the Cisco IOS software for routers and switches [9]. The main functionality of NetFlow is to export the IP information that the router equipment may possess, aggregated in flow records. The basic architecture of the NetFlow has three components: the NetFlow Data Export, the NetFlow Flow Collector (NFC), and the NetFlow Data Analyzer (NDA).

NetFlow's main component is the NetFlow Data Export. It operates inside the routing equipment. It captures and stores traffic under flow records. Periodically, it exports this information to the NFC, which collects the data. The NetFlow Data Export can aggregate the flow information under programmable flow definitions in order to reduce the amount of data to be transmitted. Among other information, it accounts for the number of packets and the number of bytes that belong to the same flow. The IP addresses, transport protocol, and port numbers define the basic flow key-identifier. As flow detection is performed inside the router, additional information belonging to the routing process is able to be stored, *e.g.*, logical interface number, next-hop router, or AS information. Once a flow expires, it will be packed with other flows, into a UDP packet to be transmitted to the NFC. UDP is faster than TCP but can lead to a loss of data in congested networks. There is, however, overhead traffic on the network that could affect the traffic under monitoring. Also, the router should reserve more resources to store NetFlow information and must send additional packets. This makes NetFlow a possible bottleneck for monitoring with high traffic load. The remaining two components in the NetFlow Architecture are the NetFlow Flow Collector and the NetFlow Data Analyzer, which collect and analyze the data exported by one or more pieces of equipment running NetFlow.

Concerning the potential performance degradation by applying NetFlow in an operational system, this is measured usually by comparing the "no drop rate" of a network with and without NetFlow being enabled. According to the documentation the no drop packet rate is degraded by about 44% in a test of switching 64 byte packets, when NetFlow is enabled on a RSP2 with 128 MB of RAM.

3.2.2 CoralReef

CoralReef is a set of tools developed by the CAIDA group (Co-operative Association for Internet Data Analysis) [33]. Its main functions are traffic capture in high-speed networks, data storage and IP traffic analysis. CoralReef allows traffic capture at different layers, *e.g.*, cells/frames, packets, flows.

Passive traffic monitoring is the main difference between NetFlow and CoralReef. CoralReef performs passive traffic monitoring using optical splitters. No router or switch is performing the capture and analysis. Instead, another computer, which receives a copy of the traffic does all the work. Consequently, no additional data have to be sent by the router and no additional resources need to be reserved inside the

network equipment. CoralReef can report information on layer two (cells/frames), layer three (IP packets) and layer four (IP flows). IP flow information is the one that reaches the highest rate of capture and data reduction in full traffic analysis. CoralReef accounts for data from each flow register it has seen in the network. The IP addresses, transport protocol, port numbers, and timestamp for the first and last packet identify a flow register. The information accounted for is the number of bytes and the number of packets belonging to that flow. One can define a maximum time between packets to detect flow expirations, or force flow expiration whenever one deems it necessary.

3.3 Security

QoS inherently includes security issues. As security services are more and more applied to Internet services, they require certain resources and induce costs. In order to set up appropriate pricing for them, tangible means have to exist. This section decomposes existing security services into intrinsic components that can be used for the quantitative management of quality of security services. This forms the technical basis for services to be charged for.

3.3.1 Security Services Taxonomy

Within the context of QoS, security related issues became a topic of research only recently. Assuring security requires proper management of resources, which consequently results in certain costs. To address these issues, appropriate concepts, i.e. taxonomy for quality of security services (QoSS) is needed. Next, these services have to be decomposed into intrinsic components and cost metrics has to be associated with each of these components. Such metrics form the basis for effective utilization of resources, pricing policies, and charging. Firstly, all variety of security services is decomposed into intrinsic components (i.e. the QoSS taxonomy). Secondly, with each of these components, appropriate cost metrics are associated to enable practical implementations. A similar approach is found in [23], but extended here by an explicit treatment of coupled issues. It should be mentioned that a Public Key Infrastructure (PKI) is considered as well, which is a problem especially for the wireless world owing to extensive computation for certificates and revocation lists [31].

With regards to taxonomy, it is essential to refer to security services using a well-established framework, which is the ISO framework [25]. It defines the following security services: authentication, confidentiality, integrity, non-repudiation, access control, and auditing/alarms. These services are implemented with various security mechanisms that can be grouped as follows:

- Cryptographic primitives (sym. and asym. algorithms, one-way hash functions);
- Access control mechanisms;
- Auditing and alarm mechanisms.

Although quite extensive, these groups are not sufficient. Experiences have shown that the availability of communication systems is very important, especially in the light of growing denial of service attacks [20]. Next, traffic flow confidentiality is also important, but it is almost completely left out from the above standards. Finally,

physical security should be included; all equipment and cryptographic keys have to be properly stored and physically protected. Thus, basic groups are given in the relation as of Table1.

The meaning of fields is as follows: *s-id* is a unique identifier of a security service, *sec-srvc* is the category of security service, and *s-name* is a descriptive name of security service. Other attributes are related to the cryptographic protocol overhead, which are: *ent-id* for identification of entities, *t-nonces* for time-stamps or nonces, *cont* for a content, *cert* for certificates and *CRL* for certificate revocation list. These attributes, measured in bytes, are building blocks of cryptographic protocols. The last attribute is measured in seconds. It is needed to calculate bandwidth usage that is an important issue when PKI related operations for mobile handheld devices are considered. Non-repudiation is left out as it is a compound service, which consists of authentication and integrity.

Table 1: Services Relation with Cost-related Attributes

s-id	sec-srvc	s-name	ent-id	t-nonces	cont	cert	CRL	time
at#	authentication	...	...	...	...	...	...	...
cf#	confidentiality							
it#	integrity							
tf#	traffic flow							
ac#	access control							
av#	availability							
ad#	audit/intrusion detection							
ps#	physical security							

3.3.2 Cost Metrics

Having taxonomy of security services, we can divide them further down to intrinsic components. This first requires identification of mechanisms, which can be later decomposed into generic operations. The definition of mechanisms is given in the relation of Table 2, where *m-id* stands for a unique identifier of a mechanism, *mechanism* for a category of a mechanism, and *m-name* for the name of a mechanism.

Access control mechanisms should not include only access lists and matrixes look-up operations, but also complex firewall operations (payload scanning). Similarly, auditing and alarm mechanisms should not include only pure event logging and related man-power, but also subsequent analysis (deployment of specialized intrusion detection techniques). From security point of view, a whole communication system should be divided into processing (active) components and transmission media (passive) components. This decomposition differs from the one described in [9], which distinguishes between intermediate nodes, end systems, wires and the total area

Table 2: Mechanism Relations

s-id	m-id	mechanisms	m-name
...	sa#	symmetric algorithm	...
	aa#	asymmetric algorithm	
	hf#	hash function	
	dp#	dummy packets	
	rr#	rerouting	
	lu#	matrix /look-up	
	cs#	code-scanning	
	sc#	software-correctness	
	bu#	bandwidth use	
	la#	og-analysis	
	pm#	physical mechanisms	

subnet. We don' t find such division of services area useful - location as described in the above mentioned work is of a little relevance to costs. Services have to be provided end to end. If there are many segments, it makes no sense to apply, *e.g.*, authentication only to end system without having authenticated the rest of the systems along the path.

Services have to be identified exactly along the communication path. It is a handy approach to relate a QoSS component to an IP address of a device, where a service takes place. Through IP numbers all CPUs and memory elements an be uniquely identified and the same holds true for transmission media, be it a wire or wireless.

There is certainly a problem of measuring QoSS with regards to availability. Although not mentioned in ISO standards, it became a serious concern in recent years, as availability is mostly degraded by (distributed) denial of service attacks [20]. The reason for these kinds of attacks is increasingly complex software. It is hard to quantify and directly relate software correctness to QoSS taxonomy and pricing. A possible objective metrics for this purpose is to use Common Criteria [24], which address this issue through EAL leveling - the more rigorous tests of equipment, the higher the price. So, denial of service is implicitly addressed through EALs. Finally, it is possible to identify basic cost components and their measures. These are as follows:

- CPU usage, measured in cycles for performing load / store, arithmetic, logical / shift, control and byte manipulation operations;
- Memory allocation, measured in bytes;
- Bandwidth usage measured in bytes per second;
- Software correctness measured in EAL levels;
- Manpower measured in man-years;
- Costs of physical protection of equipment in currency units per year.

CPU and memory usage are obtained using SNMP measurements. Bandwidth usage is obtained through amount of transferred data bits (bytes), divided by the upper time limit for a protocol to finish. Manpower and costs of physical protection are obtained as statistical aggregates during a particular period; they are not calculated repeatedly for each QoSS request.

Table 3: Cost Relation with m-id as a Foreign Key

		cost elements					
ip-id	**m-id**	**cpu**	**mem**	**sw**	**mp**	**phys**	**bw**
ip_1	sa_1	$c_{1,1}$	$c_{1,2}$	...		$c_{1,5}$	$c_{1,6}$
...							
ip_n	pm_h	$c_{n,1}$	$c_{n,2}$	...		$c_{n,5}$	$c_{n,6}$

In the above table *ip-id* presents a unique identifier, an IP address of a device (in case that a device more than one IP number, the one with the lowest value is taken). Further, *cpu* stands for CPU usage, *mem* for memory usage, *bw* for bandwidth usage, *sw* for software correctness, *mp* for manpower and *phys* for costs of physical security. It should be noted that $n \geq k$. As the amount of data should cover arbitrarily large messages, it is necessary to normalize cost elements. Except bandwidth that is computed based on SERVICES relation, and can be treated as a fixed cost for a given protocol, all other elements are thus normalized per byte of payload. To calculate the total cost C for a required set of security services, a weighted sum of cost elements $c_{i,j}$ is used and multiplied by the total amount of data D in bytes, for which security service is required:

$$C = D * \left(\sum_{i=1}^{n} \left(\sum_{j=1}^{5} c_{i,j} * w_{i,j} \right) + c_{i,6} \right), \qquad c_{i,j}, w_{i,j} \in \mathbb{R}$$

Fig. 1. Calculation of total Cost

3.3.3 Example

The following example is based on a hybrid authentication and key distribution protocol [36]. *A* and *B* stand for names of communicating parties, *E* for an encryption with a public key, while E^{-1} for decryption, and *K* for a session key:

- Bob sends to Alice his public key.
- Alice's system generates random session key, encrypts it with Bob's public key and sends it to Bob, i.e. $E_B(K)$.
- Bob decrypts by using his private key and recovers session key, i.e. $E_B^{-1}(E_B(K))=K$.

In order to function properly, it is necessary to verify public keys and it is assumed that public key infrastructure is available. Suppose that Bob wants to establish a secure session with Alice at IP address 128.231.3.4 being on the same segment as

Bob's computer. He decides for strong authentication using RSA with 512 bit long keys.

Table 4: Example of Services Relation

s-id	sec-srvc	s-name	ent-id	t-nonces	cont	cert	CRL	time
at#	authentication	hybrid with key exchange	100	0	16	3000	30000	2
...	...							

Table 5: Example of Mechanism Relation

s-id	m-id	mechanisms	m-name
at_3	aa_4	asymmetric algorithm	512-bit RSA
...	...		

Table 6: Cost Relation Example

		cost elements					
ip-id	m-id	cpu	mem	sw	mp	phys	bw
128.231.3.4	aa_4	20	100	0.005	0.001	0.002	16558
...							

Only the first two steps of the above protocol are relevant for QoSS, as the third step is done by Bob's system. Values in the relation above are obtained as follows: entity identification for Bob and Alice requires 100 Byte, there are no nonces, payload is a session key of 16 Byte, X.509 version 3 certificate has on average 3000 Byte, CRL is expected to be 30,000 Byte. Upper-bound time limit is two seconds. The value for CPU usage is 20 cycles per Byte. Further, memory usage is 100 per Byte, bandwidth usage is 33,116/2 Byte/s. EAL for the appropriate encryption module is 0.005 per Byte, human resources costs are 0.001 man years per Byte, and physical security is 0.002 currency units per Byte. Except for bandwidth usage, all calculations are related to 16 Byte payload processing, which represents secure session key. The absolute value for all weights is taken to be 1, while their units are obtained as reciprocal values of corresponding cost elements. Thus, the total cost C for establishment of a secure session is 18,478.128.

4 User-Centered Charging

Further aspects of the way users perceive a given service should be investigated, to really keep people as the focal point of service provisioning. With the convergence of voice and data services in IP networks, together with future services and applications,

the number of service providers greatly increases. Each end-to-end Internet session or transaction may involve many service providers, each one claiming a kind of payment to regenerate the consumed resources (including investments and revenue). Moreover, users may ask services from application, content, and service providers, Web servers and pages, access networks, and alike. The collection and processing of the charging data from the Internet soon becomes unmanageable for the service providers and the Internet users alike. The number of vendors and service providers operating in the market offer the average user too much choice, without any of the required help and assistance to act as a guide to what is available in the Internet. To manage this, the user is likely to want to deal with only one home ISP, which acts as a broker and mediator between the various service providers involved. Single unified billing for the services and applications consumed in the Internet is one route to improve the global QoS the users and ISPs experience. This is where the Unique-Kiosk Approach (UKA) is positioned.

The UKA bases on the Internet AAA architecture. It provides both the Internet user and service provider with a single Point-of-Charging (PoC) that allows all Internet access charges to be made from one service provider. The UKA is not tied to specific charging models or tariffs that service providers set for the service provision. To capture and process the charging data required by the UKA, an open Charging Platform (CP) architecture is designed. The UKA and CP aim to meet a basic set of requirements for charging and accounting for Internet usage:

- Provide a call detail record for all charges incurred and requiring settlement between the various commercial providers in the loop;
- Allow end users control over the charges being made by the provider;
- Allow itemized billing for all services charged to each subscription, including voice-based and data-containing phone calls, and covering services and content;
- Allow billing consolidation and convergence;
- Provide fraud detection and prevention in the network.

Fig.2 shows possible charging points that may be used by an ISP.

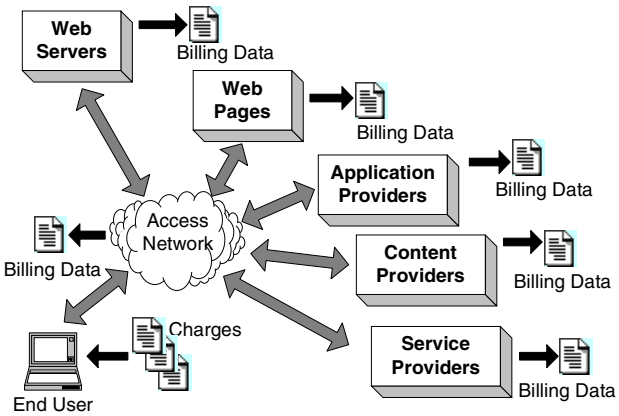


Fig. 2. Charging Points for Internet Usage

4.1 Unique Kiosk Approach

Typically, between a user and an ISP there is a communication path belonging to a network provider (NP). Thus, a user should deal, based on separate contracts, with at least two, usually different, providers, and take care of all those contracts. Moreover, in case of Internet link failure, using multiple vendors usually results in finger pointing, delays, and widespread frustration.

To address and prevent these kinds of problems, we now consider the model of the UKA. Although a service provision mechanism, UKA supports and improves the charging and billing for Internet services. In UKA, a user is to deal with only one provider by means of a two-part Virtual Kiosk (VK). The user part is running on his terminal - computer or mobile device - while the ISP part is running on ISP. The VK belongs to the ISP, which is the parent ISP (PISP) of that VK and a preferred ISP for the user. Relying on the UKA, also an ISP may become "user" for another ISP. To provide services, a PISP should open a VK on the user's terminal. Whenever the user asks a PISP for a service, the appropriate local VK obtains the necessary information from the host user, "finds and hires" the intermediate actors able to connect the user's terminal to the PISP, and further supports the requested service, while maintaining LOG files. A level of QoS is negotiated. The PISP assumes all the costs for the requested services. At the agreed interval, the PISP issues a bill containing all previous collected costs for the services the user requested and consumed. The UKA implies and supports rather than enforces charging/billing/payment policies, thus allowing freedom for the ISPs in this area. The UKA is independent of the access network. The VK knows about the communications interfaces available on the user's terminal since installation or a later re-configuration, and spawns the appropriate communication agents. Before or during an Internet session, and according to user indications, the VK may choose between NPs based on the required quality of transfer, the physical link availability and quality, tariffs at that time, in addition to other metrics.

At the installation time, the VK asks and assists the user to provide a valid client ID (CID), including *username*, *password*, and *network access identifier*. Also, a VK comes with a Kiosk ID (KID) unique for its PISP. In the two-step log in, the VK checks off-line with the user provided CID and, if it passed, connects to the PISP and sends the pair (CID, KID) for further authentication, authorization, and accounting. Except link problems, more on-line log in attempts may indicate a fraud attempt. Besides CID and KID, a VK encapsulates and hides from other parties the PoPs of its PISP and the strategy to access them (*e.g.*, using metrics like distance and time-of-day) so that the fraud is severely limited.

4.1.1 Points of Charging and Charging and Billing Information Flow

An important component of UKA is the Point-of-Charging (PoC), which is the point where the charging data, bills, or both go either for on-line or latter payment. Although a user is required to specify only one PoC, he has the freedom to specify more permanent or temporary PoCs, instructing the VK how to use them, *e.g.*, a credit card company for local payments in mobile access, the called party, in a VoIP session, or the PISP for Web access. Whenever the user accesses the Internet, the VK presents a PoC as credential indication.

Based on SLAs it has with its neighbor service and network providers, an ISP will be charged by its neighbors for every ISP's user access they support. Each provider, whose network the user traffic passed over, registers a per-flow bill entry and passes the bill backward to the appropriate neighbor provider, at the agreed interval or cycle. This way, the bills accumulate in a scalable manner at the preferred ISP. The ISP then pays to its neighbors the received bills and issues user bills to the indicated PoCs, based on the LOG files maintained by the corresponding VKs.

4.2 Mobility and Roaming

The development of mobile terminals with multiple physical or software-defined interfaces is expected to allow users to seamlessly switch between a wide variety of wired and wireless access technologies, often with overlapping areas of coverage and dramatically different cell sizes. A need has been generated to allow users to negotiate a Point-of-Attachment (PoA) to any domain convenient to their current location [1]. The VK may change the current PoA and interface used to connect to an ISP for a competing NP in its area, based on dynamic VKs.

An (access) NP may host VKs of several ISPs willing to sell their services in that area. The NP collects/distributes packets from/to users of a hosted ISP and issues a bill to that ISP for the provided service. An ISP may have both active VKs, where their user-part is already running on user terminals, and inactive VKs, where their user-part, hosted by that ISP, is waiting to migrate to a user terminal. For mobility purposes, the user-part of a VK has the capability to host the VK user-part of a foreign ISP. After the user terminal finds a PoA that best meets the user's needs, the selected ISP becomes temporarily the preferred ISP of the user, and a VK user-part migrates to and runs as guest VK user-part on the user terminal. The guest VK asks the hosting VK or the user for ID data and PoC, performs all required registrations, and supports the service.

4.3 Charging Platform Architecture

The (CP) enables the UKA to be prototyped and evaluated in live and experimental networks. Using a layered approach, the CP is concerned with the collection of traffic data for the purposes of charging, cost allocation, billing, capacity and trend analysis, and auditing. It covers the functionality of the PoCs, meeting the requirements of the AAA model. The main CP's components are:

- *Multi-User Environment and SQL Database Storage* to process the captured data into billing information, while applying multiple tariff structures and charging models, and as a tool to implement and evaluate charging models and tariffs;
- *Web Server Interface* that uses dynamically produced web pages, and allows to easily navigate, download or forward to additional back-end systems, for further processing or report generation, the CP's functionality, data and information.
- *Scripting Language Support* for rapid prototyping of new ideas and concepts with low development overhead.

Similar architectures have been proposed in [42] and [16]. Suitably extended, the CP carries out also authorization and admission control functionality, based on [1].

4.3.1 Charging Platform

On a first level, the CP acts as a mediation device, as in [27]. The added value comes in terms of the services and applications that can be used for the complex manipulation and modelling of the captured data. The CP has a flexible architecture that offers many benefits and functionality through the on-line and off-line modelling and processing of the captured data. The functionality includes:

- *Charging Data Capture* from many sources, such as: IP level traffic data, http protocol data, GSM voice and data traffic, as well as non-standard application level data. Important elements of the CP are the use of standard data formats, for compatibility with other platforms in use, and the granularity of the data captured, as the level to which network traffic can be metered and charged.
- *Charging for Content* that requires the capture of sufficient data to enable detailed analysis of what content has passed over the network. Instructed by PISP, a VK provides the CP with all relevant data for the user's degree of satisfaction.
- *Charging Models and Tariffs*: Providing a flexible framework for implementation of various charging models, the CP can capture, archive, and process charging data of a wide range of formats. The CP is also designed to model new charging models on archived charging data. Such modelling and analysis will hopefully result in more cost-effective and innovative charging schemes.
- *Charging* for QoS, as a necessary feature of most networks that provide guaranteed service levels, and as a metric that users are willing to pay a premium for. The CP supports differential charging, using suitable charging models.
- *Scalability and Growth:* The CP can run on a single host or on host systems distributed in the network, to reduce the total throughput and capacity requirement on each host, and the traffic carrying the metering information. The number of distributed CPs is likely to increase much more slowly than the number of users.
- *Meeting the Internet Charging Requirements,* as summarized in Table 1.

Table 7: Internet Charging Requirements

Charging Requirements	UKA and Charging Platform Architecture Solution
Call and session records	Produced using the captured and processed network data
User control over charging	UKA allows the user choice and control over the costs via selecting suitable ISP, PoC, and network resources
Itemized billing	Produced using the captured and processed network data
Billing convergence	'One-stop shop' approach of the UKA and the data processing carried out by the charging platform
Fraud detection	Implemented in combination with standard AAA architectures; UKA hides the sensitive data of an ISP from unknown users

4.4 Conclusions

Internet access is usually based on services and equipment from different providers. Many vendors usually result in finger pointing and general frustration. To avoid this, the UKA performs on behalf of a preferred ISP all the activities the Internet connection requires. This approach gives the user great flexibility in choosing where to procure Internet services from, while keeping a centralized payment arrangement. The use of the CP architecture to capture and process the network data required by the UKA aims at unified Internet charging [11], on metrics over and above the usual access and duration measurements, and simplicity in the charging function.

5 Content Charging

The use of Internet technology for content distribution in a closed or controlled Intranet is an evolving business model for content providers. This allows the use of Internet protocols for the delivery, exchange and distribution of the content and also the implementation of mechanisms for improved or configurable QoS. Charging for the content and the associated delivery is required to enable the various content and network providers involved to generate revenue. Total charge for content is made up of the transport cost plus the cost or charge for the actual content. The cost of transport is made up of the Internet access charges plus access charges when accessing or traversing peering core/access networks plus the charge for the QoS required for the connection. Subscription models are very suited to charging for content delivery from both a consumer and provider viewpoint and may be implemented with current protocols and technology.

5.1 Content Distribution and Exchange

Within the last five years content and its electronic distribution has become more and more important for the communications and media industry. This trend is driven by a number of developments. On the consumer side it is the possibility to receive digital content for instance via the Web or Digital Video Broadcasting (DVB). In the production domain it is the increase of digital platforms and production formats in conjunction with the demand for an ever-increasing amount of quality information that has to be delivered quicker than before.

5.1.1 What Is Content?

According to the SMPTE (Society of Motion Picture and Television Engineers) and EBU (European Broadcasting Union) task force definition [38] content consists of essence and metadata. Essence is the raw programme material itself and represents picture, sound, text, or video. Metadata describes the actual essence and comprises all kinds of data related to the actual essence. Metadata includes content related metadata that describes the actual content; material related metadata that describes the available formats and location related metadata describing the location, number of copies and condition of carrier. Depending on the application area the two different constituents of content are of varying relevance. Metadata is needed to describe and find content. The actual essence is consumed and operated upon.

5.1.2 Content Distribution Requirements

Applications require support for content location, communication and charging. On top of this there might be a number of special services that support for instance the distribution of processes, media analysis and indexing, localized branding. In the context of this section the focus is on communication and charging aspects.

5.1.3 Communications Requirements

Content has by definition multiple parts, viz. metadata and essence. The metadata is conventionally composed of discrete media elements such as text and images, i.e. key frames. However, metadata segments can have a time relationship, especially when they are referring to segments within a piece of continuous media. The essence of continuous media is clearly time dependent. Very often it is composed of multiple components, *e.g.*, a video track and a number of audio tracks, that have to be combined and displayed at the same time. Content is usually delivered either by streaming or file transfer. A mixture of the two delivery mechanisms that involves caching is also possible. Media streaming is especially in the context of live events required. In order to assure a continuous data flow QoS mechanisms have to be in place. If content is delivered via file transfer very large files have to be handled. These files contain not only essence but also metadata.

5.1.4 Charging Requirements

Only very few goods can be exchanged electronically and content is one of the major products in this area. This implies that there are two chargeable elements in the exchange and delivery of content, viz. the communication costs and the costs for the content itself, i.e. the rights to consume, publish, use. Because of the high bandwidth and QoS requirements the cost of communication is substantial. Content costs depend on the kind of usage. Content distribution or xCast might be charged in a similar way to cable and satellite TV. Video-on-Demand charges are usually modelled similar to video rental prices. Costs for the acquisition of content IPRs (Intellectual Property Rights) highly depend on the kind of content, how many rights holders there are and the kind of rights that will be acquired. There can be a rather complex negotiation process involved. Both parts of content charging can be done entirely separately or charging models can be developed that combine communication and content costs.

5.2 Charging for Content Delivery within CDNs

For charging and billing to take place within a CDN suitable accounting systems need to be in place that can monitor and record network events associated with request-routing, distribution, and delivery of the digital content. As these network events are normally in the upper OSI layers (4-7) it may be preferable to use accounting protocols and methods that are also hosted in those layers. This would abstract the accounting tasks away from the low level TCP/IP transport of the underlying Internet, and should reduce the amount of accounting data collected for typical content distribution transactions. Charging and billing across a CDN requires infrastructure to authenticate users, create usage records, mediation and aggregation of the usage data, pricing of the usage and content, and consolidation and settlement of charges across various peering networks. Accounting systems must also be able to scale to size,

reach and complexity of the peering CDN and not add heavyweight performance overheads to networks.

5.2.1 Service Level Agreements (SLAs) for CDNs

SLAs need to be in place with all negotiated relationships that the CDN operator maintains to be able to offer different levels of QoS to content providers and users alike. A framework for such SLAs has been proposed for Premium IP networks [16], which may also be applied to a CDN overlaid onto the Internet. When SLAs are in place then effective network measurement should also be implemented to ensure that all parties concerned are adhering to the SLAs and charging can be adjusted accordingly. The measurements required should include as a minimum congestion monitoring, failed content requests and end-to-end delays within the CDN.

5.2.2 The Role of MPEG Standards in Billing and Charging for CDNs

There are several relationships that can be identified between the IETF CDI (Content Distribution Internetworking) [21] and ISO/IEC MPEG initiatives. Possibly the most obvious refers to the usage of the MPEG standards for the representation of the coded multimedia content that can be exchanged between, and delivered by, CDNs. The mutual relevance between these initiatives extends much beyond content representation, as exampled by the most recent ISO/IEC MPEG initiative with the Multimedia Framework or MPEG-21 [29], [26]. The MPEG-21 initiative aims at assuring interoperability in the interactions that can occur in the entire multimedia chain (creation/distribution/usage/consumption) [5]. The seven technical areas that are considered as relevant for this interoperability correspond to the six key core qualifiers that can be applied to user pair interactions plus the reporting of events within these interactions [4], [18]. Both initiatives consider at the same time charging and billing systems as out of their standardization scope and identify accounting (event reporting) as the cornerstone that has to be standardized in order to implement these systems [5]. Furthermore, both initiatives recognize that the definition of a standardized interface and set of metrics are the key components of interoperable accounting systems. CDN accounting system metrics are expressed and conveyed via *Content Detail Records* (CDRs). The payload of a CDR is composed of several fields that qualify in a quantitative/unambiguous manner the activity that is being accounted. The MPEG-21 standard also aims to address different technical requirements that are associated with the representation and distribution of CDRs.

5.2.3 Subscription Charging for Content Delivery

Using subscription services the content delivery provider becomes a broker for the content being distributed and delivered from the content providers. Content delivery/distribution providers may be able to set up subscription services to the content they host that covers the basic connectivity charges as well as content charging, and act as an ISP. Alternatively a content delivery only subscription service is possible without the added requirements of ISP provision. For subscription charging to work a layered model is required to allow revenue generation to be achieved across a wide customer base. Such layered models have already been successfully implemented in the cable and satellite TV markets, as well as the printed media market for some years. Subscription models have so far proved to be

sustainable and successful business models for companies such as B-Sky-B, AOL-Time-Warner, and Kingston Communications. Internet provision is a richer media than cable and satellite TV so the metrics that may be charged for in a subscription are also more diverse. The Internet is capable of carrying and delivering a wide range of services and content and this is reflected in the layered subscription model. QoS provision based on priority scheduling or on bandwidth division or resource reservation is possible based on and dependent upon the level of subscription purchased by the end-user. Increased requested priority results in higher subscription charge for the content being delivered, or possibly varying the quality of the delivered source content, or possibly the scheduling of time-sensitive content, *e.g.*, live/recorded football matches. Other value added content and or meta-data may be chargeable through this model, *e.g.*, MPEG-4 type object content, varying camera angles on streaming video for live sport events etc. A layered subscription model may include the following elements:

- Flat-rate subscription for Internet access, and basic services such as e-mail and web browsing. This may be purchased direct from the ISP or content delivery provider as a package. Un-metered Internet access is becoming the standard business model in this space, and makes sense for customers and providers alike, providing the take up rate is sufficient to cover the overheads.
- Internet access purchased may be based on one or multiple access technologies including dial-in modem, ISDN, xDSL, WLAN, and GSM/GPRS/UMTS.
- Alternatively the Internet access may be purchased from other access providers.
- Subscription packages for valued added services that can be customized to the customers' requirements. Such services may include video, audio, and event content, such as MTV, music/video on demand and news channels.
- Subscription packages for pay-per-view content, such as movies on demand, sporting events and news articles. The emphasis here is on higher value content or better quality content that may be charged for reasonably by the provider.
- Subscription packages for QoS on demand, which allow the customer to have a budget for priority and bandwidth allocation requests over the network for Internet connection and content delivery. This would also allow the customer to purchase more QoS budget if/when required to compensate for expected network usage and utility.

Past research [3], [28] has shown that Internet users like to have predictable charges for their Internet access. Subscription models fulfil this user requirement since the level of spending can be controlled by the level of subscriptions taken out. The layered subscription model also provides the content providers and the network providers with value added revenue streams that subscribers can choose to take up or not. This allows the baseline subscription revenue to be supplemented by the package add-ons and extras. Once subscribers are signed up to the content delivery service converged payments may be used to pay for all the services purchased through one vendor, thereby simplifying the invoicing, charging and payment for both subscriber and provider alike. The charges and subscription rates imposed for the various layers of the subscription model are set by the provider according to their business model, current market trends and the cost of any sourced content and delivery infrastructure.

5.3 Conclusions

Content distribution and exchange is the key to the future development of the Internet. Infrastructure and methods for the auditing, accounting and charging for the interchange are necessary to enable content delivery and exchange to be efficiently charged for and to enable the required revenue generation and collection.

The total charge for the distributed content needs to be a reasonable charge for the content being delivered based on the value of the content and the delivery mechanism employed. Subscription based charging reduces the accounting and billing overhead for the content providers and network providers alike. It also has the advantage of prepayment for content being delivered or consumed. A layered subscription model as proposed provides more choices for the consumer and also the possibility of increased revenue streams for the content and network providers. Both consumers and providers mutually benefit by offering configurable QoS as a user selectable option or parameter in a pricing plan that also has an associated charge. This enables content providers and network providers to generate the revenues required to sustain, maintain and grow the Internet and the services offered.

6 Case Study on Cost Sharing

The case study discussed here is based on the experience acquired in the MIRA project [43]. This project was focused on developing a simple, but effective, charging scheme based on volume and traffic classification for high-speed Internet links. The system bases byte accounting and byte classification on four characteristics of each byte: its origin, destination, application and direction. This charging scheme is more complex than simple byte accounting, and may result in fairer billing and provides additional information about user profiles.

The objective of the traffic classification method for billing is to use heuristics in order to discern user behavior (group of applications, origins, and destinations) from layer 3 and layer 4 header information. Once each byte is accounted for under a class of traffic, the user can be charged for it, taking into account the potential academic use, which includes applications for research, teaching, and development purposes.

The work is oriented towards traffic charging and billing based on network resource usage for private groups (NRNs, corporate networks) that currently offer free Internet access or flat rate access and wish to continue offering services in a fairer way. Network resource usage is characterized by the volume of the traffic and the potential academic profile. The academic profile is derived from traffic classification combinations (group of applications and origins and destinations).

In the particular case of the Spanish NRN (RedIRIS) the topology shows characteristics which made it easy to apply a charging scheme. The topology is a star, with a central node, managed by RedIRIS, and one link for each political administrative region/nationality. Moreover, three additional links connect the RedIRIS Backbone to the Internet. One link connects RedIRIS to the European R&D network Géant. Another link connects RedIRIS to the Spanish Internet Exchange Point (Espanix) where commercial Network Providers are also connected. Finally, a third link connects RedIRIS to the United States for the traffic directed there or for

those routes that do not have a path through the Espanix or Géant Networks. Traffic going to the United States may be either American or default-route traffic.

As there are no loops in this topology, monitoring it link by link provides the proper information (no duplications) of the traffic in the Spanish NRN. The regional access points do not inject transit traffic from third-party networks. Consequently, all the traffic present in the regional links belongs to RedIRIS-connected institutions. In order to distinguish the local chargeable institutions, each is assigned a network prefix. To characterize the traffic by what link that traffic uses to gain access to a institution or to go to a destination, external entities are characterized by an Autonomous System group number. Therefore, we can characterize the destination/origin of each traffic flow with a 4x17 matrix, as there are seventeen institutions, one representing each link; and 4 destinations, one for each group of external networks. Each flow detected in a link is also characterized by its application port. Sets of ports join together applications with the same characteristics. Seventeen groups of applications are currently defined.

Finally, a charging matrix can be programmed in order to charge combinations of origin/destinations, applications, and traffic direction. Each origin/destination can be charged differently depending on the potential academic use of that link. Every application group can be charged relative to the potential academic use of applications.

6.1 Traffic Accounting Platform

The basic requirement is that the capture platform must be passive. This requirement allows the parallel development of a traffic capture and analysis system without affecting network performance and administration.

At time the of this study the high-speed transport technology in the Spanish NRN was ATM155. A pair of optical splitters supplied the passive copy of the traffic. Each splitter gives a copy of the incoming and the outgoing traffic of the link under study. The accounting platform can be deployed in the central node, accounting entirely all the regional networks, or it can be deployed at the regional access point. In this case, it can account for smaller institutions. A combination of both configurations was used in order to account for the central node and the regional node for Catalonia separately.

A full-rate traffic accounting platform was deployed using CoralReef flow-based accounting tools, which reports flows instead of packets, thus reducing the amount of information to be analyzed for traffic classification. Now, traffic classes are based on volume, origin, destination, traffic direction, and application (determined from the TCP/UDP ports in the transport header). Later, the academic nature of each traffic flow is set by the traffic classification processes based on some heuristics.

6.2 Traffic Classifications

The IP flows are translated into charging flows by adding class attributes. Subsequently, they are logged under the same flow record, taking into account only the new class attributes. The number of flows is reduced drastically. The source and destination values are reduced to MxN, where M is the number of institutions under consideration, and N is the number of sets of destinations defined for the traffic flows.

Also, the number of applications was reduced from the maximum TCP/UDP port number to the number of defined application classes.

These processes are executed once every charging period. The charging period depends on the minimum time for which one wishes to apply a charging scheme. Four class attributes have been selected: the traffic direction, local institution, application group and destination group. Except for the direction class, all class attributes can be configured by adding or removing attributes from configuration files. In this way, the platform is flexible and can be deployed easily in other networks.

Traffic direction: Each traffic flow is classified independently, by taking into account its direction. By maintaining different records for incoming and outgoing traffic, different charging schemes can be applied to each direction of the traffic.

Institution: The second class attribute is the local chargeable institution to which the flow belongs. An identifier represents each institution. There is a database where identifiers are related to network prefixes. As such, institutions having several network prefixes with the same label are charged under the same charging scheme. Also, you can define highly-detailed institutions by defining long network prefixes. The main rule to determining which institution a flow belongs to is by finding which one has the longest prefix match. Each flow is characterized by the institution attribute after analyzing the source IP address (for outgoing traffic) or the destination IP address (for incoming traffic). In the regional node for Catalonia, 40 institutions have been selected for charging. For a centralized charging scheme 17 institutions were defined, one for each regional network. In this case, the IP source or destination addresses are not analyzed, but instead an institution is assigned to each flow based on its VPI/VCI label. As each regional node has a unique VPI/VCI identifier, this method allows for greater speed in the traffic analysis process.

Location: The location class identifies the external origin or destination of the traffic (from the point of view of the Spanish NRN). In order to reduce the number of choices and to relate the information to an academic criterion, four destination attributes are defined. Each attribute represents a set of Autonomous Systems. All Autonomous Systems in a group are connected to a different external link in the core NRN. From a NRN point of view, each set represents a different type of network. The first group is the Spanish NRN group itself, which represents traffic between Spanish institutions in different regional Networks. The second group is the Géant group, which represents traffic to other European NRNs. The third group is the Espanix group, which represents commercial networks in Spain. The last group is the USA group. This group is the default route for the traffic, and represents most of the traffic going to the United States or going to other networks not connected via Geant or Espanix. In spite of the type or usage of the applications, these destinations have, a priori, more information about the academic nature of the traffic.

Application: The application class attribute is derived from the analysis of the source and destination ports in TCP/UDP headers. Other important transport protocols with no port information, such as ICMP, are also classified. Applications are classified into groups based on the potential academic usage of each application. In this case seventeen application groups are defined.

6.3 Charging and Billing

There are 5,440 different ways of combining the different types of bytes to be charged in the regional node for Catalonia (40 institutions, 2 directions, 4 locations and 17 application groups). Moreover, if we want to apply different charging matrices for different time periods (work week, weekend and day/night), the number of combinations increases. The application class attribute is derived from the analysis of the transport protocol and the source and destination ports in TCP/UDP headers, according to the well-known and registered port numbers, and the non-standard ports used by the most known applications. Other important transport protocols with no port information, such as ICMP, are also classified. Applications are classified into groups based on the potential academic usage of each application. In this case seventeen application groups are defined.

Table 8: Charging Matrix

Location/Application	**E-mail**		**Games**		**P2P**		**WWW**		**Unknown**	
	In	***Out***	***In***	***Out***	***In***	***Out***	***In***	***Out***	***In***	***Out***
RedIRIS	0.1	0.1	3	4	4	5	0.2	0.2	1	1
GÉANT	0.2	0.2	3	4	4	5	0.3	0.3	1	1
ESPANIX	1	1	3	4	4	5	1.5	1.5	1	1
USA (default)	0.5	0.5	3	4	4	5	0.6	0.6	1	1

Table 8 shows an example of a possible traffic charging matrix. Four applications are shown in order to reduce space. Unknown application traffic is charged based only on volume; no variations between locations are applied. Other well-known applications have different charges. For example, the price of e-mail increases from 0.10 € per gigabyte for within the Spanish NRN to 1.00 € to Spanish commercial networks. Nevertheless, e-mail service may be considered to be academic, and the price rises depending on the link cost, with one exception. Although a local link to a Spanish commercial network may be cheaper than a link to USA, the use of the mail service in this environment may be considered to be less academic. Other applications clearly out of the scope of academic usage are charged heavily. The charges are higher for outgoing traffic than for incoming traffic, in order to charge more for the service of a non-academic application than for access to that type of service.

This matrix is applied to volume classifications each charging period. Fig. 3 determined the incoming traffic classification volume for one entity over the course of one day. Each application class has four Location columns. Fig. 4 shows the incoming traffic costs once the charging matrix is applied. For the outgoing traffic the results are similar.

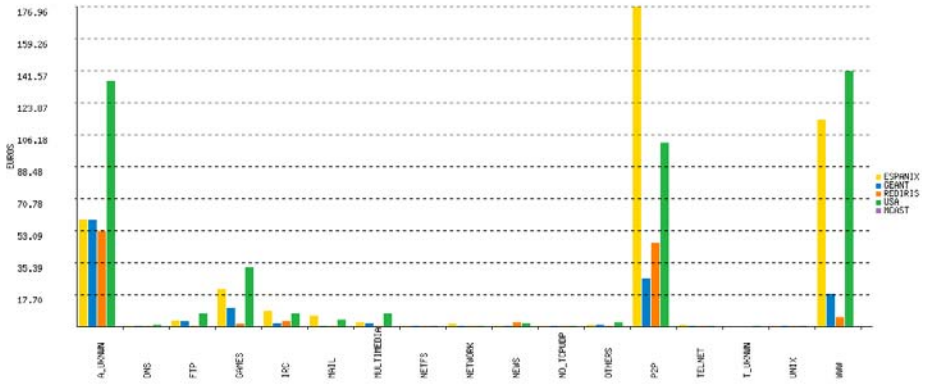


Fig. 3. Incoming Traffic Classification Costs

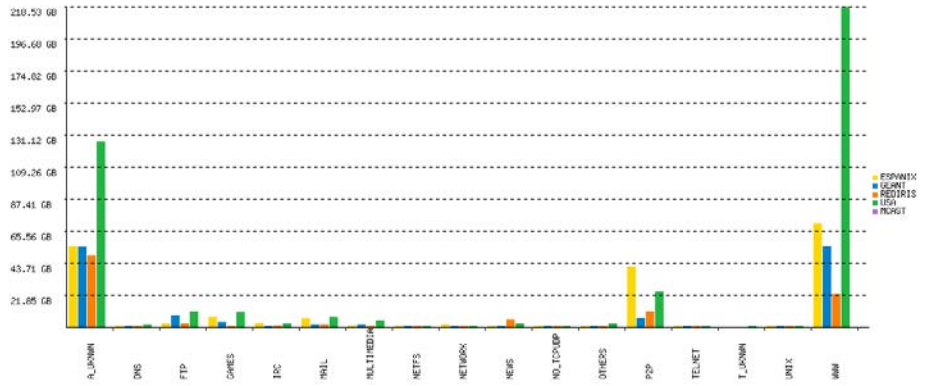


Fig. 4. Incoming Traffic Classification Volume

6.4 Conclusions

Full high-speed traffic analysis in real time with great detail is possible with low-cost resources. The tested traffic accounting and classification platform for cost-sharing in the Spanish NRN was developed exclusively using standard PC equipment. In addition, the traffic-capture tasks do not degrade the network performance, because they are passive. Therefore, the platform can be installed in current networks, and it gives enough information to update some current charging and billing systems for NRNs, since most of them are based only on traffic volume accounting.

Besides the information needed by the billing process, much valuable traffic information is collected during the traffic accounting and classification processes. This information resides in flow registers and is not used for billing, but for gathering knowledge on institution traffic profiles and usage of network resources. Also, it permits to detect irregular usage or attacks on the studied network and to apply traffic engineering in order to optimize traffic costs.

The transmission speed of network technologies is growing. The backbone network speed increases and the routing/switching process will be faster than real-time analysis. Therefore, full traffic accounting for high-speed links will become unfeasible with low cost resources. To keep costs and resources for analysis low, a sampling method should be applied in the future. A preliminary study on this topic can be found in [44].

7 Overall Conclusions and Outlook

This chapter on pricing and Quality-of-Service in the Internet addressed views of providers and users. While traditional user-provider relationships are defined by consumer and seller roles, more advanced pricing models take into account that neither the provider nor the user dominate the Internet. However, the technology in place, the mechanisms applied, the charging schemes offered, and the QoS delivered to the user define provider-centric tasks. As shown in this chapter, those key components linked into a suitable, applicable, and efficient technology platform, configured by economic goals, and addressing user needs are feasible, though a little complex today to operate.

Therefore, a dedicated and important task set can be outlined, identifying those areas of research, provider support, and user behavior investigations, which demand further work and studies. On the technology side and on the economic part, advances have to be addressed: Appropriate protocols and interfaces have to be defined and standardized to enable the use of cost metrics, which are linked into the communication system to serve user perceived QoS, security, AAA extensions, traffic and accounting analysis, and control algorithms. Research into economically viable charging models and their technically efficient implementations define an important goal of this research, where those control algorithms combine technically measured parameters with business model driven economic targets. A balanced combination of those parameters will allow for a successful user discrimination in the market of Internet Service Provider offering content and various other Internet services.

The authors like to thank Lars Wolf, TU Braunschweig for valuable discussions and Jörg Diederich, TU Braunschweig for managing parts of the editorial process.

8 References

1. B. Aboba et al.: *Criteria for Evaluating AAA Protocols for Network Access;* RFC 2989, http://www.ietf.org/rfc/rfc2989.txt, November 2000.
2. B. Aboba, D. Mitton (Chairs): *IETF AAA WG;* http://www.ietf.org/html.charters/aaa-charter.html, March 2003.
3. J. Altmann: *How to Charge for Network Services - Flat-Rate or Usage-Based?* Special Issue on Networks and Economics, Computer Networks Journal, August 2001.
4. J. Bormans, K. Hill (eds): *Study on MPEG-21 (Digital Audiovisual Framework) Part 1 v2.0;* Document ISO/IEC JTC1/SC29/WG11/N4040, Singapore, March 2001

5. J. Bormans, Keith Hill (Ed.s): *MPEG-21 Overview;* Document ISO/IEC JTC1/SC29/WG11/N4511, Pattaya, December 2001.
6. B. Briscoe, V. Darlagiannis, O. Heckman, H. Oliver, V. Siris, D. Songhurst, B. Stiller: *A Market Managed Multi-service Internet (M3I);* Computer Communications, Vol 26, No. 4, March 2003, pp 404-414.
7. N. Brownlee: *New Zealand Experiences with Network Traffic Charging;* ConneXions, Vol. 8, No. 12, 1994.
8. CADENUS project: *Creation and Deployment of End-User Services in Premium IP Networks;* http://www.cadenus.org/, March 2003.
9. Cisco Systems: *NetFlow Services Solutions Guide;* http://www.cisco.com/univercd/cc/td/doc/cisintwk/intsolns/netflsol/nfwhite.pdf, 2001.
10. D. Clark: *A Model for Cost Allocation and Pricing in the Internet;* in L. W. McKnight, J. P. Bailey (Eds): Internet Economics; MIT Press, Cambridge, Massachusetts, U.S.A., 1997.
11. J. Cushnie, D. Hutchison, M. Popa: *Internet Charging using the Unique-Kiosk Approach;* COST263 Technical and Management Meeting, Namur, Belgium, http://www.comp.lancs.ac.uk/computing/users/cushniej/jc_papers.html, December 2001.
12. B. Day, K. Hoadley (Consultation Paper): *Network Funding;* http://www.ukerna.ac.uk/archive/response_jisc_399.html, 1999.
13. M. Day, D. Gilletti, P. Rzewski: *Content Internetworking (CDI) Scenarios;* Internet Draft (work in progress), <draft-ietf-cdi-scenarios-00.txt.pdf>, February 2002.
14. M. Day, B. Cain, G. Tomlinson, P. Rzewski: *A Model for Content Internetworking (CDI);* Internet Draft (work in progress), <draft-ietf-cdi-model-01.txt>, February 2002.
15. T. Dolan: *Internet Pricing. Is the end of the World Wide Wait in View?* Communications & Strategies, Vol. 37, 2000, pp 15-46.
16. ETSI - European Organization for Standardization: *NA8 GONOW;* http://www.etsi.org/
17. M. Falkner, M. Devetsikiotis, I. Lambadaris: *An Overview of Pricing Concepts for Broadband IP Networks;* IEEE Communications Surveys, Second Quarter 2000.
18. A. Ganesh, K. Laevens, R. Steinberg: *Congestion Pricing and User Adaptation;* IEEE Infocom, Anchorage, Alaska, U.S.A., April 2001.
19. R. Gibbens, F. Kelly: *Resource Pricing and the Evolution of Congestion Control;* Automatica, Vol. 35, 1999, pp 1969-1985.
20. J. K. Houle, G. M. Weave: *Trends in Denial of Service Attack Technology;* CERT Coordination Center, 2001.
21. IETF: *Content Distribution Internetworking (CDI) Working Group;* http://www.ietf.org/html.charters/cdi-charter.html, March 2003.
22. C. Irvine, T. Levin: *Toward a Taxonomy and Costing Method for Security Services;* 15th Annual Computer Security Applications Conference, Phoenix, December 1999.
23. C. Irvine, T. Levin: *Toward Quality of Security Service in a Resource Management System Benefit Function;* Heterogeneous Computing Workshop, May 2000.
24. ISO: *Common Criteria for Information Technology Security Evaluation - Security Assurance Requirements;* ISO / IEC 15408, Geneva 1999.
25. ISO, Information Technology, Open Systems Interconnection: *Security Frameworks in Open Systems;* IS 10181 - 1 through 7, Geneva 1995.
26. ISO/MPEG Requirements Group: *Current Vision on Event Reporting in MPEG 2;* Document ISO/IEC JTC1/SC29/WG11/N5338, Awaji, December 2002.

27. ITU: *International Telecommunication Union;* http://www.itu.int/, March 2003.

28. S. Jagannathan, K. Almeroth: *Price Issues in Delivering E-Content on Demand;* ACM SIGCOMM Exchanges, May 2002.

29. R. Koenen: *From MPEG-1 to MPEG-21: Creating an Interoperable Multimedia Infrastructure;* Document ISO/IEC JTC1/SC29/WG11/N4518, Pattaya, December 2001.

30. J. MacKie-Mason, H. Varian: *Pricing the Internet;* in B. Kahin, J. Keller (eds): Public Access to the Internet, Prentice Hall, 1995.

31. S. K. Miller: *Facing the Challenge of Wireless Security;* IEEE Computer, July 2001, pp 16-18.

32. MobyDick project: *Mobility and Differentiated Services in a Future IP Network;* http://www.ist-mobydick.org, February 2003.

33. D. Moore, K. Keys, R. Koga, E. Lagache, K. Claffy: *The CoralReef Software Suite as a Tool for System and Network Administrators;* http://www.caida.org/outreach/papers/2001/CoralApps/CoralApps.pdf, 2001.

34. A. M. Odlyzko: *The History of Communications and its Implications for the Internet;* http://www.research.att.com/~amo/history.communications0.ps, 2000.

35. C. Rensing, H. Hasan, M. Karsten, B. Stiller: *AAA: A Survey and Policy-based Architecture and Framework;* IEEE Network Magazine, Vol. 16, No. 6, November/December 2002, pp 22-27.

36. B. Schneier: *Applied Cryptography;* John Wiley, New York, U.S.A., 1996.

37. S. Shenker, D. Clark, D. Estrin, S. Herzog: *Pricing in Computer Networks: Reshaping the Research Agenda;* Telecommunications Policy, Vol. 20, No. 3, 1996, pp 183-201.

38. SMPTE/EBU Task Force for Harmonized Standards for the Exchange of Program Material as Bitstreams: *Final Report: Analyses and Results;* http://www.smpte.org/, 1998.

39. B. Stiller, J. Gerke, P. Reichl, P. Flury: *A Generic and Modular Internet Charging System for the Cumulus Pricing Scheme;* Journal of Network and Systems Management, Vol. 3, No. 9, September 2001, pp 293-325.

40. B. Stiller, J. Gerke, P. Reichl, P. Flury: *Management of Differentiated Service Usage by the Cumulus Pricing Scheme and a Generic Internet Charging System;* 7th IEEE/IFIP Integrated Network Management Symposium (IM 2001), Seattle, Washington, U.S.A., May 2001, pp 93-106.

41. B. Stiller, P. Reichl, S. Leinen: *Pricing and Cost Recovery for Internet Services: Practical Review, Classification, and Application of Relevant Models;* Netnomics - Economic Research and Electronic Networking, Vol 3, No. 2, 2001, pp 149-171.

42. SUSIE Project: *Charging for Premium IP Services;* http://www.teltec.dcu.ie/susie/, 2003.

43. C. Veciana Nogués, J. Domingo Pascual, J. Solé Pareta: *Cost-sharing and Billing in the National Research Networks: the MIRA Approach;* Terena Networking Conference. Limerick, Ireland, June 3-6, 2002.

44. C. Veciana Nogués, A. Cabellos Aparicio, J. Domingo Pascual J. Solé-Pareta: *Verifying IP Meters from Sampled Measurements;* In: Schieferdecker, I.; König, H.; Wolisz (Eds.): IFIP 14th International Conference on Testing Communicating Systems (TestCom 2002). Berlin: Kluwer Academic Publishers, 2002, pp 39-54.

Author Index