Graphenalgorithmen: Perfekte Graphen

Teilnehmerskript zu einer Vorlesung
von Stefan Felsner
im Wintersemester 1995/96
an der Freien Universität Berlin

3. April 1996

Dieses Skript wurde als Übung von den Teilnehmern der Vorlesung erstellt. Ich habe jedes Kapitel um einen Abschitt erweitert in dem die Quellen und weiterführende Literatur angegeben werden. Obwohl ich die Mitschriften korrekturgelesen habe sind sie höchstwahrscheinlich nicht völlig fehlerfrei. Wir hoffen, daß das Skript dem Leser trotz all seiner Mängel von Nutzen sein wird.

Stefan Felsner

Die Teilnehmer

und die von ihnen betreuten Vorlesungen

Bernd Falkenberg	**11, 19**
Carola Wenk	**5, 17**
Carsten Ihlemann	**9, 16**
Christof Schultz	**6, 18**
Clemens Hendler	**8, 15**
Falk Haferkorn	**3, 13**
Helmut Weil	**4, 14, 22**
Johannes Schwarz	**12, 21**
Stefanie Gerke	**2, 10**
Susanne Gebauer	**7, 20**

Inhaltsverzeichnis

1 Graphen und Färbungen

1. Vorlesung, 19. Oktober 95
Mitschrift: Stefan Felsner

1.1 Definitionen und Beispiele

Ein *Graph* $G = (V, E)$ besteht aus einer Menge V, deren Elemente *Knoten* genannt werden, und einer Menge $E \subseteq \binom{V}{2} = \{\,\{u, v\} : u \neq v\,;\, u, v \in V\,\}$, deren Elemente *Kanten* genannt werden. Meist wird die Größe $|E|$ mit m bezeichnet. Wenn nicht explizit darauf hingewiesen wird, gehen wir in dieser Vorlesung immer von einer endlichen Knotenmenge V aus. Die Kardinalität $n = |V|$ heißt die *Ordnung* des Graphen.

Ein Graph $G = (V, E)$ heißt *vollständig*, wenn jede mögliche Kante in G ist, d.h. $E = \binom{V}{2}$. Der *Komplementärgraph* eines Graphen $G = (V, E)$ ist der Graph $\overline{G} = (V, \overline{E})$ mit $\overline{E} = \binom{V}{2} - E$. Der *vollständige Graph der Ordnung* n wird üblicherweise mit K_n bezeichnet.

Wir sagen, die Kante $e = \{u, v\}$ *verbindet* u und v. Oft schreiben wir verkürzt $e = uv$ statt $e = \{u, v\}$. Wenn $v \in e$, dann nennen wir v und e *inzident*. Wenn uv eine Kante des Graphen G ist, dann heißen u und v *benachbart* in G. Die *Nachbarschaft* $N(v)$ eines Knotens $v \in V$ ist die Menge der benachbarten Knoten, d.h. $N(v) = \{\,u : uv \in E\}$. Der *Grad* eines Knotens v ist $d_v = |N(v)|$. Der *Maximalgrad* $\Delta(G)$ ist der maximale in G vorkommende Grad, formal $\Delta(G) = \max_{v \in V} d_v$. Der Begriff der *Nachbarschaft* wird auf Teilmengen U der Knotenmenge erweitert $N(U) = \bigcup_{u \in U} N(u)$.

Beobachtung. (i) $\sum_{v \in V} d_v = 2m$
(ii) Die Anzahl der Knoten ungeraden Grades ist in jedem Graphen gerade.

Wir nennen $G' = (V', E')$ einen *Untergraphen* (auch *Subgraphen*) der Graphen $G = (V, E)$, wenn $V' \subseteq V$ und $E' \subseteq E$ und schreiben dann $G' \subseteq G$. Wenn G' alle Kanten von G enthält, die Knoten aus V' verbinden, dann ist G' der von V' *induzierte Subgraph* $G' = G[V']$.

Eine Folge $v_0, v_1, \ldots, v_k$ von Knoten ist ein *Weg der Länge* k von v_0 nach v_k in $G = (V, E)$ wenn $v_{i-1}v_i \in E$ für $i = 1, .., k$. Ein Weg W heißt *einfach* wenn kein Knoten mehrfach in W vorkommt, einfache Wege werden manchmal auch *Pfade* genannt.

Eine Folge $v_0, v_1, \ldots, v_k, v_0$ von Knoten ist ein *Kreis der Länge* $k + 1$ in

$G = (V, E)$ wenn $v_{i-1}v_i \in E$ für $i = 1, .., k$ und $v_k v_0 \in E$. Ein Kreis heißt *einfach*, wenn $v_i \neq v_j$ für $i \neq j$. Eine Kante $uw \in E$ ist *Sehne (chord)* eines einfachen Kreises $v_0, v_1, \ldots, v_k, v_0$ wenn $u = v_i$ und $w = v_j$ für $|i - j| \geq 2$. Der einfache sehnenlose Kreis der Länge n wird als Graph mit C_n bezeichnet.

Ein Graph $G = (V, E)$ heißt *zusammenhängend* , wenn es zwischen je zwei Knoten von G einen Weg gibt. Ein maximaler zusammenhängender Subgraph von G ist eine *Komponente* von G. Eine Kante e heißt *Brücke* wenn $G_e = (V, E - e)$ mehr Komponenten hat als G. Ein *isolierter Knoten* ist eine Komponente ohne Kanten. Einen Graphen ohne einfache Kreise nennt man *Wald*, ein zusammenhängender Wald ist ein *Baum*.

Eine Menge $I \subseteq V$, für die $G[I]$ nur aus isolierten Knoten besteht, heißt *unabhängig*. Mit anderen Worten: I ist unahhängig, wenn es in I keine benachbarten Knoten gibt. $\alpha(G)$ ist die Kardinalität einer größten unabhängigen Menge in G.

Lemma 1. *Es gilt* $\alpha(G)(\Delta(G) + 1) \geq |V|$.

Beweis. Sei I unabhängig mit $|I| = \alpha(G)$. Wenn es ein v gibt mit $v \notin I \cup N(I)$, dann ist $I \cup v$ eine größere unabhängige Menge — Widerspruch. Also ist $V = I \cup N(I)$. Mit $|N(I)| \leq |I|\Delta(G)$ folgt die Behauptung. □

Übung 1. *Es gilt* $\alpha(G) \geq \sum_{v \in V} \frac{1}{d_v + 1}$.
Hinweis: Konstruiere eine unabhängige Menge durch wiederholte Wegnahme des Knotens kleinsten Grades und seiner Nachbarn. Betrachte die Veränderung des Gewichts $Phi(G) = \sum_{v \in V} \frac{1}{d_v + 1}$ durch so eine Wegnahme.

Eine *gute t-Färbung* ist eine Partition der Knoten $V = X_1 + X_2 + .. + X_t$, so daß jedes X_i eine unabhängige Menge ist. Wie denken uns die Knoten aus X_i mit der "Farbe" i gefärbt und stellen fest, daß benachbarte Knoten mit verschiedenen Farben gefärbt sind. Färbungen sind in unserem Zusammenhang fast immer gute Färbungen. Die *chromatische Zahl* $\chi(G)$ von G ist das kleinste t zu dem eine gute t-Färbung von G existiert.

Lemma 2. *Es gilt* $\chi(G) \leq \Delta(G) + 1$.

Beweis. Sei $v_1, .., v_n$ eine Aufzählung der Knoten von G und $c_0, .., c_\Delta$ eine Aufzählung der "Farbpalette". Färbe v_1 mit Farbe c_0. Seien $v_1, .., v_{k-1}$ schon gefärbt. Für v_k wählen wir die kleinste Farbe mit der noch kein Nachbar von v_k gefärbt ist. Da v_k höchstens Δ Nachbarn hat, unsere Palette aber $\Delta + 1$ Farben enthält, finden wir immer so eine Farbe in unserer Palette. □

Bemerkung. Der Algorithmus, den wir im Beweis des Lemmas zum Färben von Graphen verwendet haben, ist unter den Namen ListColoring oder GreedyColoring bekannt. Er wird uns im Laufe der Vorlesung noch öfter begegnen.

Übung 2. [Satz von Brooks] *Sei G ein zusammenhängender Graph. Es gilt $\chi(G) \leq \Delta(G)$ außer wenn es ein n gibt, so daß $G = K_n$ oder $G = C_{2n+1}$.*

Lemma 3. *Für jeden Graphen G gilt $|V| \leq \alpha(G)\chi(G)$.*

Beweis. Sei $t = \chi(G)$ und $X_1, .., X_t$ eine Partition von V in t unabhängige Mengen. Es gilt $|V| = |X_1| + .. + |X_t|$ und $|X_i| \leq \alpha(G)$, also $|V| \leq t\alpha(G)$. □

Übung 3. *Für jeden Graphen G gilt $|V| \leq \chi(G)\chi(\overline{G})$.*

Eine Menge $A \subseteq V$ für die $G[A]$ ein vollständiger Graph ist heißt *Clique*. Wenn die Größe r einer Clique betont werden soll, so spricht man kurz von einer *r-Clique*. $\omega(G)$ ist die Kardinalität einer größten Clique in G.

Eine *Cliquenüberdeckung* ist eine Partition der Knoten $V = Y_1 + Y_2 + .. + Y_s$, so daß jedes Y_i eine Clique in G ist. $\kappa(G)$ ist die kleinste Anzahl von Cliquen mit der eine Cliquenüberdeckung von G möglich ist.

Wir fassen zwei einfache Beobachtungen im folgenden satz zusammen.

Satz 4.

(1) *Für jeden Graphen G gilt $\omega(G) \leq \chi(G)$ und $\alpha(G) \leq \kappa(G)$.*

(2) *$\omega(G) = \alpha(\overline{G})$ und $\chi(G) = \kappa(\overline{G})$.*

Beispiele. Einige wichtige Graphen, die uns noch öfter begegnen werden:

K_n der *vollständige* Graph auf n Knoten, auch *n-Clique* genannt.

C_n der sehnenlose Kreis auf n Knoten, auch *n-Kreis* genannt. Der 3-Kreis $C_3 = K_3$ heißt auch *Dreieck*.

P_n der sehnenlose Pfad auf n Knoten, er entsteht aus C_n durch das Entfernen einer Kante, auch *n-Pfad* genannt.

rK_n r disjunkte Kopien von K_n.

$K_{m,n}$ der Komplementärgraph der disjunkten Vereinigung von K_m und K_n. Diese Graphen heißen *vollständig bipartite Graphen*. Der Graph $K_{1,n}$ heißt auch *Stern*.

Ein Graph $G = (V, E)$ heißt *bipartit*, wenn es $m, n \in \mathbb{N}$ gibt, so daß G ein Subgraph von $K_{m,n}$ ist. Die Knotenmenge eines bipartite Graphen kann in zwei unabhängige Mengen zerlegt werden. Will man so eine Zerlegung $V = X_1 + X_2$ hervorheben, so schreibt man $G = (X_1, X_2, E)$.

Übung 4. *G ist bipartit, genau wenn $\chi(G) \leq 2$.*

1.2 Drei Anwendungen von Färbungen

Beispiel 1

Sei X eine Menge von Studentinnen die an einem Tag alle zuerst eine Veranstaltung an der FU und später eine Veranstaltung an der TU besuchen. Für Studentin x endet die FU-Veranstaltung um a_x und beginnt die TU-Veranstaltung um c_x, die Reisezeit zwischen FU und TU ist R. Die Studentinnen reisen bevorzugt in grösseren Gruppen.

Sei $b_x = c_x - R$, dann muß Studentin x die FU zu einem Zeitpunkt t_x aus dem Intervall $[a_x, b_x]$ verlassen. Studentinnen x und y können genau dann gemeinsam in einer Gruppe reisen, wenn $[a_x, b_x] \cap [a_y, b_y] \neq \emptyset$. Definiere einen Graphen G auf X: Als Kanten wählen wir genau die Paare $\{x, y\}$ für die $[a_x, b_x] \cap [a_y, b_y] \neq \emptyset$ gilt. Offenbar ist jede Gruppen Y, die gemeinsam reisen kann, eine Clique in G. Die Menge der Gruppen überdeckt X, also ist $\kappa(G)$ eine untere Schranke für die Anzahl von Gruppen in denen die Studentinnen die Reise durchführen. Gilt Gleichheit? Die affirmative Antwort beruht auf der folgende Aussage.

Behauptung. Jede Clique kann gemeinsam reisen.

Beweis. Sei Y eine k-Clique in G. Sei a_y der Größte bei Elemeten aus Y auftretende Intervallanfang. Für $z \in Y$ gilt nun: $a_z \leq a_y$ und $[a_z, b_z] \cap [a_y, b_y] \neq \emptyset$, also ist $b_z \geq a_y$ und $a_y \in [a_z, b_z]$. Damit ist gezeigt, daß a_y ein zulässiger Reisetermin für alle Elemente von Y. △

Bemerkung. Eine Familie $\{A_i\}_{i \in I}$ von Teilmengen einer Menge A erfüllt die *k-Helly Eigenschaft* wenn $J \subseteq I$ und $\bigcap_{j \in K} A_j \neq \emptyset$ für alle $K \subseteq J$ mit $|K| = k$ schon $\bigcap_{j \in J} A_j \neq \emptyset$ impliziert. Wir haben soeben die 2-Helly Eigenschaft der Intervalle nachgewiesen. Allgemein gilt: Konvexe Mengen in Dimension d erfüllen die $(d+1)$-Helly Eigenschaft.

Ein Graph $G = (V, E)$ heißt *Durchschnittsgraph (intersection graph)* einer

Familie $\{B_v\}_{v \in V}$ von Teilmengen von B, wenn $\{u, w\} \in E \iff B_u \cap B_w \neq \emptyset$. Der Graph in unserem Beispiel war der Durchschnittsgraph einer Familie von Intervallen, diese Graphen heißen *Intervallgraphen.*

Übung 5. *Zeige, jeder Graph ist als Durchschnittsgraph darstellbar.*

Übung 6. *Finde ein Verfahren, mit dem die Studentinnen eine optimale Zerlegung in Gruppen bestimmen können.*

Beispiel 2

Ein zentraler Post-Paket-Umschlagplatz besteht aus einer langen Halle. Auf der linken Seite der Halle befinden sich Entladepositionen X und auf der rechten Seite Beladepositionen Y. Ist eine Garnitur Lastwagen auf der linken Seite angefahren, so bekommt der Hallenleiter für jedes $x \in X$ eine Liste $L_x \subseteq Y$ mit den Beladepositionen, zu denen die Pakete bei x müssen. Zu jeder Entladeposition gehört ein Wagen, der Pakete befördert. Der Umladevorgang ist in Runden organisiert. Für jede Runde gilt: (i) Der Wagen jeder Entladeposition bringt Pakete zu höchstens einem Ziel auf der Beladeseite und kehrt dann zurück. (ii) Jede Beladeposition wird von höchstens einem Wagen angefahren. (iii) Aus Sicherheitsgründen ist es verboten, daß sich die Wegeverschiedener Wagen kreuzen. Die Aufgabe des Hallenleiters ist es die jeweils nötige Anzahl von Runden zu minimieren.

Wir modellieren das Problem in einem Graphen: Als Knoten nehmen wir die Paare (x, y) mit $y \in L_x$. Die Kanten sollen Wegen entsprechen die nicht in einer Runde gefahren werden können. Zum einen sind das Paare $\{(x, y), (x', y')\}$ sich kreuzender Wege, zum anderen müssen wir, weil pro Entladeposition nur ein Wagen existiert, Paare $\{(x, y), (x', y')\}$ mit $x = x'$ berücksichtigen. Es ist leicht einzusehen, daß die chromatische Zahl dieses *Konfliktgraphen* gerade die optimale Anzahl von Runden ist.

Wir werden später sehen, daß wir für allgemeine Graphen nicht erwarten können, ein effizientes Verfahren zur Bestimmung der chromatischen Zahl zu finden. Für viele Klassen von Durchschnittsgraphen gibt es jedoch solche Verfahren. Können wir für obigen Graphen eine "einfache" Darstellung als Durchschnittsgraph finden?

Tatsächlich legt unser Anwendungsbeispiel so eine Darstellung nahe. Es ist klar, daß sich die Knoten (x, y) des Graphen G so durch Strecken mit einem Endpunkt auf einer senkrechten Geraden g_X links von einer senkrechten Gerade g_Y, auf der der zweite Endpunkt liegt, darstellen lassen, daß sich kreuzende Wege in der Halle sich kreuzenden Strecken entsprechen. Verlegen wir die linken Endpunkte der Strecken ein wenig nach links auf die Gerade $g_X - \epsilon$ und die rechten Endpunkte ebenso nach rechts auf die Gerade $g_Y + \epsilon$, so haben wir unseren Graphen G als Durchschnittsgraphen von Strecken mit Endpunkten auf zwei parallelen Geraden dargestellt.

Die Darstellung ist nun so, daß keine zwei Strecken mehr einen Endpunkt gemeinsam haben. Identifiziere die Knoten von G mit $1,..,n = |V|$, so daß die Endpunkte der Strecken auf $g_X - \epsilon$ in der Reihenfolge $1,..,n$ von unten nach oben auftreffen. Liest man die Namen der Strecken auf $g_Y + \epsilon$ von unten nach oben, so erhält man eine Permutation π von $1,..,n$. Die Kanten von G sind genau die Paare $\{i,j\}$ mit $i < j$ und $\pi_i > \pi_j$.

Ein Graph $G = (V,E)$ ist ein *Permutationsgraph* wenn es eine Bijektion $f : V \to \{1,..,n\}$ und eine Permutation π gibt, so daß $\{v,w\} \in E \iff f(v) < f(w)$ und $\pi_{f(v)} > \pi_{f(w)}$.

Der Hallenleiter steht also vor dem Problem, optimale Färbungen von Permutationsgraphen zu bestimmen.

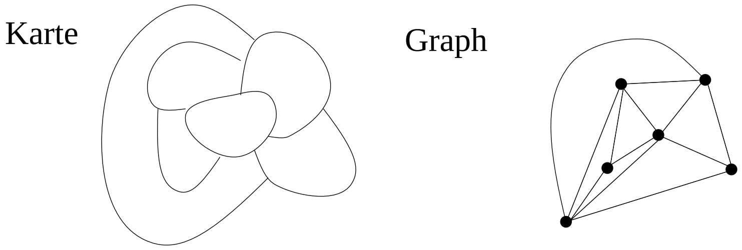
Übung 7. *Das Komplement eines Permutationsgraphen ist wieder ein Permutationsgraph.*

Ein Graph $G = (V,E)$ heißt *containment Graph* einer Familie $\{B_v\}_{v \in V}$ von Teilmengen von B, wenn $\{u,w\} \in E \iff B_u \subseteq B_w$.

Bemerkenswerterweise sind Permutationsgraphen genau die containment Graphen von Familien von Intervallen. Beweisskizze: Die eine Richtung ist sehr einfach. Gegeben ein Permutationsgraph auf n Knoten, hänge zwei Kopien von $1,..,n$ als $1_1,..,n_1,1_2,..n_2$ hintereinander und ordne dem Knoten i das Intervall $[i_1, \pi(i)_2]$ zu. Für die Umkehrung sei eine Familie $\{A_i\}_{i \in I}$ von Intervallen gegeben. Ordne die Indexmenge I so, daß die linken Endpunkte aufsteigend sortiert sind. Die Ordnung der rechten Endpunkte definiert die gesuchte Permutation π.

Beispiel 3

Gegeben sei eine Landkarte. Die Länder der Landkarte sollen so gefärbt werden, daß Länder mit einer gemeinsamen Grenze verschieden eingefärbt sind. Wie viele Farben sind dafür nötig?



Diese Aufgabe läßt sich in ein reines Graphenproblem übersetzen: *Was ist die chromatische Zahl eines ebenen Graphen?*. Dieses Problem hat eine lange und aufregende Geschichte. Eine umfassende Behandlung des Problems und gleichzeitig ein historischer Abriß ist das Buch von Aigner [1].

Zum Kapitel

Mehr über Graphen und Färbungen findet man in jedem Lehrbuch zur Graphentheorie. Stellvertretend seien hier die schönen Bücher von Wilson [46] und Aigner [1] erwähnt. Das für Ende 96 angekündigte Buch *Introduction To Graph Theory* von D. West wird sicher auch sehr empfehlenswert.

Anwendungen wie Beispiel 1 und 2 finden sich zahlreich in Golumbics [18] und in Büchern von Fred Roberts. Eine historische Behandlung des 4-Farben Problems die auch Originalquellen einschliesst ist Biggs, Lloyd und Wilson [3].

2 Gerichtete Graphen und Ordnungen

2. Vorlesung, 24. Oktober 95
Mitschrift: Stefanie Gerke

2.1 Gerichtete Graphen

Ein *gerichteter* Graph $D = (V, A)$ (engl. directed graph, digraph) ist eine Menge V und eine Menge $A \subseteq V \times V$. Den Graphen $\widetilde{D} = (V, \widetilde{A})$ mit $\widetilde{A} = \{\{u, v\} : (u, v) \in A\}$ nennt man den *zugrundeliegenden* Graphen von D. Eine Folge $v_0, v_1, \ldots, v_k$ von Knoten ist genau dann ein Kreis in $D = (V, A)$, wenn die Folge ein Kreis im zugrundeliegenden Graphen von D ist. Die Folge von Knoten ist ein *gerichteter* Kreis, wenn $(v_i, v_{i+1}) \in A$ und $(v_k, v_0) \in A$. Ein gerichteter Graph heißt *azyklisch* (directed acyclic graph) wenn er keine gerichteten Kreise enthält; wir werden solche Graphen auch kurz als *Dag* bezeichnen. Bei einem gerichteten Graphen kann man den Grad jedes Knotens aufspalten in die Anzahl d^- der Kanten, die zu v hinführen und in die Anzahl d^+ der Kanten, die v verlassen. Knoten mit $d^- = 0$ nennt man *Quelle*, Knoten mit $d^+ = 0$ *Senke.*

Übung 1. *Jeder Dag besitzt eine Quelle und eine Senke.*

Folgerung. Es gibt eine Reihung $v_1, \ldots, v_n$ der Knoten eines Dags, so daß alle Kanten vorwärts zeigen, d.h. $(v_i, v_j) \in A \Rightarrow i < j$. So eine Reihung nennt man *topologische Sortierung.*

Eine binäre Relation R heißt

symmetrisch, wenn für alle $x, y; x \neq y$ gilt: $(x, y) \in R \Rightarrow (y, x) \in R$.

antisymmetrisch, wenn für alle $x, y; x \neq y$ gilt: $(x, y) \in R \Rightarrow (y, x) \notin R$.

reflexiv, wenn für alle x gilt: $(x, x) \in R$

irreflexiv, wenn für alle x gilt: $(x, x) \notin R$

transitiv, wenn für alle x, y, z gilt: $(x, y) \in R$ und $(y, z) \in R \Rightarrow (x, z) \in R$

Beispiele.

(1) Sei $G = (V, E)$ ein Graph, dann ist die zugehörige Relation R definiert durch $(x, y) \in R \iff \{x, y\} \in E$ symmetrisch. Hat der Graph keine Schlingen, so ist die Relation irreflexiv.

(2) Eine Äquivalenzrelation ist reflexiv, symmetrisch und transitiv.

(3) Eine strenge Halbordnung z.B. "$<$" ist irreflexiv und transitiv, woraus unmitellbar die Antisymmetrie folgt.

Sei R eine Relation, dann nennt man die Relation R^{-1} mit $(x, y) \in R^{-1} \Leftrightarrow (y, x) \in R$ die zu R *inverse* Relation. Ein gerichteter Graph $D = (V, A)$ wird als *orientierter* Graph bezeichnet, wenn $A \cap A^{-1}$ gleich der leeren Menge ist. Ein orientierter Graph mit zugrundeliegendem Graphen G ist eine *Orientierung* von G.

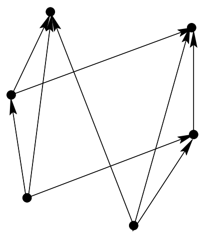
2.2 Ordnungen

$P = (V, <)$ heißt *Ordnung*, falls $<$ eine strikte Ordnungsrelation ist.

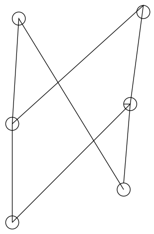
Beobachtung. Jede Ordnung ist ein Dag (gäbe es einen Kreis, so wäre die Ordnungsrelation aufgrund der Transitivität reflexiv)

Ordnungen werden graphisch auf verschiedene Weise repräsentiert.

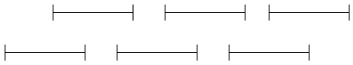
- Als Dag.



- Als *Diagramm*, d.h., ohne transitive Kanten und ohne Pfeilspitzen (Orientierung nach oben).



- Spezielle Darstellung, z.B. durch Intervalle.

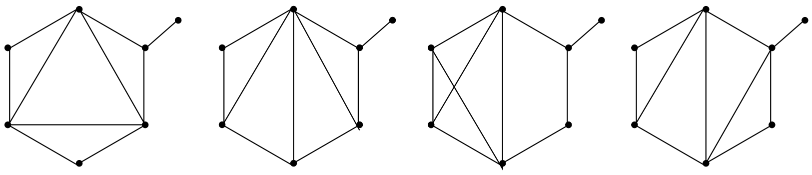


Eine Ordnung $P = (V, <)$ heißt *Intervallordnung* wenn es eine Familie $\{I_v\}$ von Intervallen gibt, so daß I_v genau dann ganz vor I_w liegt, wenn $v < w$.

Übung 2. *Finde eine Ordnung, die keine Intervallordnung ist.*

Der zugrundeliegende Graph einer Ordnung heißt *Vergleichbarkeitsgraph*, sein Komplement *Unvergleichbarkeitsgraph*. Die Unvergleichbarkeitsgraphen von Intervallordnungen heißen *Intervallgraphen*.

Übung 3. *Welche der folgenden vier Graphen sind Intervallgraphen?*



Eine *Kette* ist eine Menge von Knoten die paarweise vergleichbar sind und eine *Antikette* ist eine Menge von Knoten die paarweise unvergleichbar sind. Es gelten die folgenden Entsprechungen:

Ordnung P	$\longleftrightarrow$	G Vergleichbarkeitsgraph von P
Kette in P	$\longleftrightarrow$	Clique in G
Antikette in P	$\longleftrightarrow$	unabhängige Menge in G

Satz 1. *Die maximale Länge einer Kette in P ist gleich der minimalen Anzahl von Antiketten, in die P zerlegt werden kann.*

Beweis. Die Menge der minimalen Elemente ist eine Antikette. Entferne alle minimalen Elemente. Dadurch erhält man eine neue Ordnung, deren minimalen Elemente wieder eine Antikette in der ursprünglichen Ordnung bilden. Iteriere diese Vorgehensweise. Man bekommt eine Folge von Antiketten $A_1, \dots A_k$. Nun kann man eine Kette konstruieren, indem man ein Element x_k aus A_k nimmt, seinen Vorgänger aus A_{k-1} usw.. Ist $x_i \in A_i$ gewählt, so findet man einen Vorgänger in A_{i-1}, weil x_i ansonsten bereits vor der Entfernung der Elemente von A_{i-1} minimal gewesen wäre. So erhält man eine Kette der Größe k.

Da für jede Kette C und jede Antikettenzerlegung $\mathcal{A}$ die Ungleichung $|C| \leq |\mathcal{A}|$ gilt ist obige Antikettenzerlegung minimal und die Kette maximal. □

Satz 2. [Satz von Dilworth] *Die maximale Größe einer Antikette in P ist gleich der minimalen Anzahl von Ketten in die P zerlegt werden kann.* □

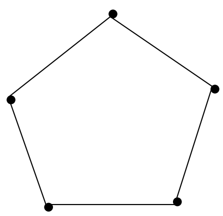
Folgerung. Für Vergleichbarkeitsgraphen G gilt stets:

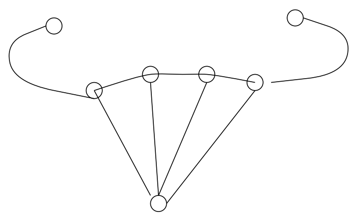
$$\omega(G) = \chi(G) \text{ und } \alpha(G) = \kappa(G).$$

Man nennt eine Orientierung, also die Zuordnung einer Richtung zu jeder Kante, so daß die zu gehörige Relation transitiv ist, eine *transitive Orientierung* des Graphen.

Übung 4. *Jeder Inklusionsgraph ist transitiv orientierbar.*

Übung 5. *Der C_5 und der BullHead sind nicht transitiv orientierbar.*





2.3 Intervallgraphen und Intervallordnungen

3. Vorlesung, 31.10.95
Mitschrift: Falk Haferkorn

Intervallgraphen sind Unvergleichbarkeitsgraphen. Für einen Intervallgraphen G und den komplementären Unvergleichbarkeitsgraphen $\overline{G}$ gilt:

$$\begin{aligned} \alpha(G) = \kappa(G) &\iff \omega(\overline{G}) = \chi(\overline{G}) \\ \omega(G) = \chi(G) &\iff \alpha(\overline{G}) = \kappa(\overline{G}) \end{aligned}$$

Beobachtung. Ist $G = (V, E)$ ein Intervallgraph und $W \subseteq V$ dann ist auch $G[W]$ ein Intervallgraph.

Definition. Sei $\mathcal{G}$ die Menge aller Graphen. Eine Abbildung $\lambda : \mathcal{G} \longrightarrow \{0, 1\}$ heißt *hereditär*, wenn aus $\lambda(G) = 1$ und H ist induzierter Subgraph von G folgt $\lambda(H) = 1$.

Bemerkung. Die Relation 'induzierter Subgraph' ist eine Ordnungsrelation $<_s$ auf $\mathcal{G}$. $<_s$ ist eine*Quasi-Wohlordnung*, d.h. jede absteigende Menge hat ein kleinstes Element. (Wohlordnung bedeutet jede Menge hat ein kleinstes Element.)

Lemma 3. *Hereditäre Grapheneigenschaften entsprechen den Antiketten in der Ordnung* $(\mathcal{G}, <_s)$.

Beweis. $\mathcal{A}$ Antikette in $(\mathcal{G}, <_s)$, definiere:

$$\lambda_{\mathcal{A}}(G) = \begin{cases} 0 & \Leftrightarrow \exists H <_s G, H \in \mathcal{A} \\ 1 & \text{sonst.} \end{cases}$$

Wir zeigen $\lambda_{\mathcal{A}}$ ist hereditär. Sei $\lambda_{\mathcal{A}}(G) = 1$ und sei $H <_s G$ für jedes $K <_s H$ gilt $K <_s G$ also $K \notin \mathcal{A}$. Damit ist auch $\lambda_{\mathcal{A}}(H) = 1$.

Sei umgekehrt λ hereditär gegeben. Wir definieren $\mathcal{A}_\lambda := \{G : \lambda(G) = 0 \text{ aber } \lambda(H) = 1 \text{ für alle } H <_s G\}$. Zu zeigen $\mathcal{A}_\lambda$ ist Antikette. Angenommen $\exists G_1, G_2 \in \mathcal{A}_\lambda$ mit $G_1 <_s G_2$. Es ist $\lambda(G_1) = 0$, weil $G_1 \in \mathcal{A}_\lambda$. Aus $G_2 \in \mathcal{A}_\lambda$ und $G_1 <_s G_2$ folgt $\lambda(G_1) = 1$. Widerspruch.

Bemerke abschließend: Ist $\mathcal{A}'$ gegeben, dann ist $\mathcal{A}_{\lambda_{\mathcal{A}'}} = \mathcal{A}'$ und ist λ' gegeben, dann ist $\lambda_{\mathcal{A}_{\lambda'}} = \lambda'$. □

Definition. Ist λ eine hereditäre Grapheneigenschaft, so nennt man die Elemente von $\mathcal{A}_\lambda$ die *minimalen Obstruktionen* für $\mathcal{A}$.

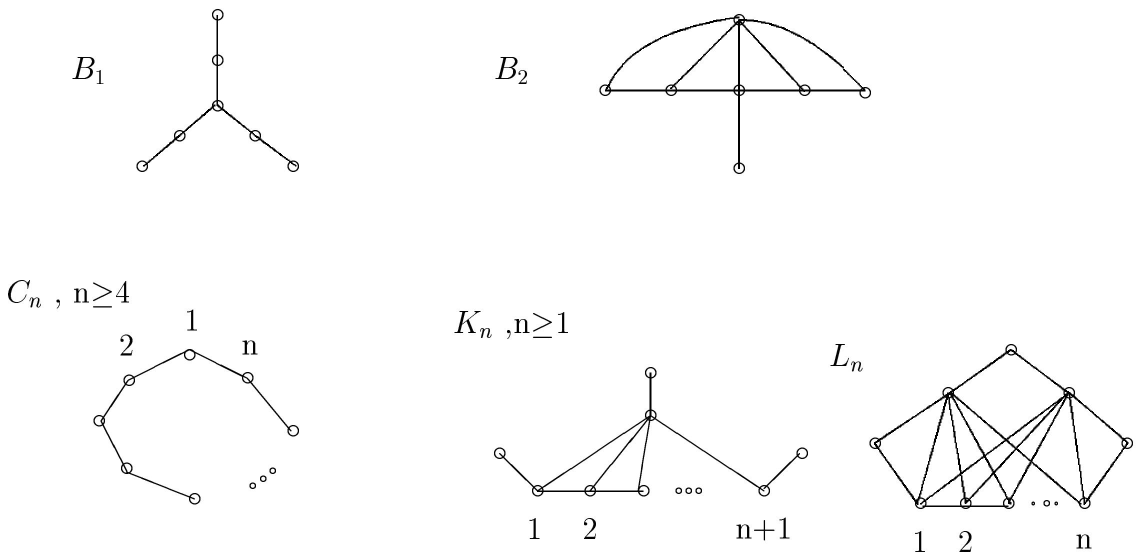


Abbildung 1: Obstruktionen für Intervallgraphen

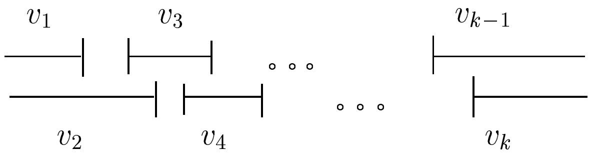
Übung 6. *Kann λ auch charakterisiert werden durch*

$$\mathcal{B} = \{G : \lambda(G) = 1 \text{ und } \forall H \text{ mit } G <_s H \text{ gilt } \lambda(H) = 0\}?$$

Bemerkung. Grapheneigenschaften deren minimale Obstruktionen bekannt sind, heißen *gut charakterisiert.* (Beispiel: Kuratowskis Charakterisierung der planaren Graphen).

Satz 4. [Lekkerker, Boland] *Die minimalen Obstruktionen gegen die Eigenschaft Intervallgraph zu sein sind genau die Graphen aus Abbildung 1.*

Lemma 5. *Sei $v_1, ..., v_k$ ein sehnenfreier Pfad in einem Intervallgraphen G $\Longrightarrow$ in jeder Intervallrepräsentation von G sind die Intervalle so*



oder umgekehrt angeordnet. Etwas formaler: Sei $I(v_i) = [a_i, b_i]$ dann genügt die Reihenfolge der Endpunkte den Ungleichungen

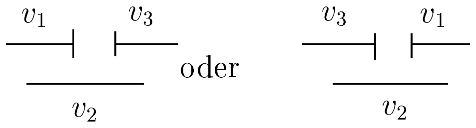
$$b_1 < a_3 \leq b_2 < a_4 \leq b_3 < a_5 \ldots a_{k-1} \leq b_{k-2} < a_k$$

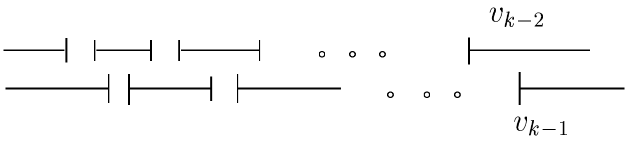
oder

$$a_1 > b_3 \geq a_2 > b_4 \geq a_3 > b_5 \ldots b_{k-1} \geq a_{k-2} > b_k.$$

Beweis. Wir führen einen Induktionsbeweis.

$k = 3$





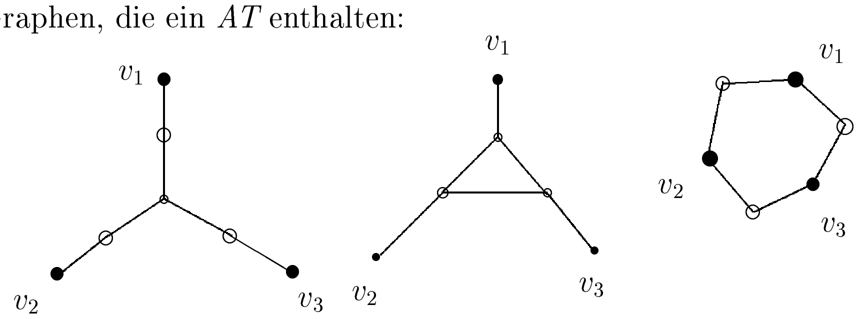
Offenbar kann das Intervall von v_k hier nur rechts angebaut werden. $\triangle$

Folgerung. In Intervallgraphen enthält jeder Kreise der Länge ≥ 4 eine Sehne. Angenommen G enthält induzierten $C_k = (v_1, ..., v_{k-1}, v_k, v_1)$. Der sehnenfreie Pfad $v_1, ..., v_{k-1}$ sieht in jeder Intervallrepräsentation so aus wie im Lemma. Das Intervall $I(v_k)$ von v_k muß mit den Intervallen von v_1 und von v_{k-1} nichtleeren Durchschnitt haben. Daher hat $I(v_k)$ auch nichtleeren Durchschnitt mit den Intervallen aller übrigen v_i. Damit induziert die Menge $\{v_1, ..., v_{k-1}, v_k\}$ aber keinen Kreis (Widerspruch). $\square$

Definition. Ein Graph heißt *trianguliert (chordal,rigid circuit)*, wenn jeder Kreis der Länge ≥ 4 eine Sehne hat. (Trianguliert ist die durch die Obstruktionsmenge $\mathcal{A} = \{C_n : n \geq 4\}$ definierte hereditäre Grapheneigenschaft.)

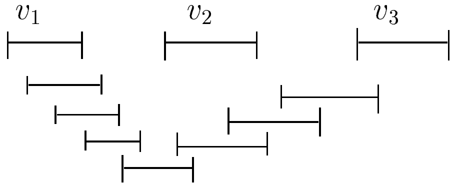
Definition. Ein *asteroides Trippel* (AT) in einem Graphen ist eine unabhängige Menge $\{v_1, v_2, v_3\}$ zu der es Pfade P_{12}, P_{23}, P_{31} gibt, so daß P_{ij} ein Pfad von i nach j ist und $P_{ij} \cap N(v_k) = \emptyset$ für $\{i, j, k\} = \{1, 2, 3\}$.

Einige Graphen, die ein AT enthalten:



Lemma 6. *Intervallgraphen sind AT-frei.*

Beweis.



Jeder Pfad von v_1 nach v_3 enthält Nachbarn von v_2. □

Satz 7. [Lekkerkerker & Boland] *Intervallgraphen sind genau die AT–freien triangulierten Graphen.*

Satz 8. [Gilmore & Hoffmann] *Intervallgraphen sind genau die triangulierten Unvergleichbarkeitsgraphen.*

2.4 Perfekte Graphen

Definition. Ein Graph $G = (V, E)$ heißt *χ–perfekt* wenn für jedes $W \subseteq V$ gilt $\omega(G[W]) = \chi(G[W])$. Wenn für jedes $W \subseteq V$ gilt $\alpha(G[W]) = \kappa(G[W])$, so heißt G *α–perfekt.*

- Die Bedingung $\omega(G) = \chi(G)$ alleine sagt nicht viel über G aus. Beispiel: Vereinigt man einen Graphen G mit n Knoten disjunkt mit K_n so hat die Vereinigung die Eigenschaft.
- χ–perfekt und α–perfekt sind hereditäre Eigenschaften.
- Vergleichbarkeitsgraphen und Unvergleichbarkeitsgraphen, also auch Intervallgraphen sind χ– und α–perfekt.

Satz 9. [*Perfect Graph Theorem*](Fulkerson, Lovász)

$$G \text{ ist } \chi\text{-perfekt} \iff G \text{ ist } \alpha\text{-perfekt.}$$

Vermutung. [*Strong Perfect Graph Conjecture*](Berge)
Die minimale Obstruktionsmenge für perfekte Graphen ist

$$\{C_{2k+1} : k \geq 2\} \cup \{\overline{C_{2k+1}} : k \geq 2\}$$

Zum Kapitel

Gerichtete Graphen und Ordnungen sind findet man in der einen oder anderen Niesche in den meisten Büchern zur Graphentheorie. Als weiterführende Literatur zu Ordnungen empfehle ich für algorithmische Aspekte Möhring [38] und für kombinatorische Aspekte Trotter [49]. Intervallgraphen und Intervallordnungen werden in den Büchern von Golumbic [18], Trotter [48] und natürlich Fishburn [13] behandelt. Die Originalreferenz zu den Sätzen von Lekkerkerker und Boland ist [29].

Die wichtigste Quelle für perfekte Graphen und ihre algorithmischen Aspekte ist zweifellos Golumbics klassisches Buch [18]. Unverzichtbar war mir außerdem der von Berge und Chvátal herausgegebene Sammelband [2] und der Übersichtsartikel von Lovász [30]. Zum gleichen Thema wie Golumbic und sogar auf deutsch gibt es ein Buch von Simon [43].

Die Vermutung von Berge stand ganz am Anfang der Theorie der perfekten Graphen, sie wurde 1960 in Halle an der Saale vorgestellt. Diese Vermutung steht auch heute noch im Zentrum des Interesses wirkt als Triebfeder der Theorie. Auch die Aussage des Perfect Graph Theorems wurde zuerst von Berge vermutet. Den Originalbeweis von Lovász aus dem Jahre 1971 findet man in [2]. Wir werden den Beweis des Perfect Graph Theorems in Kapitel 7 nachholen. Eine kurze Darstellung der Geschichte der Theorie perfekter Graphen gibt die Einleitung von [2].

3 Algorithmen und Graphen

4. Vorlesung, 2.11.95
Mitschrift: Helmut Weil

3.1 Definitionen und ein Beispiel

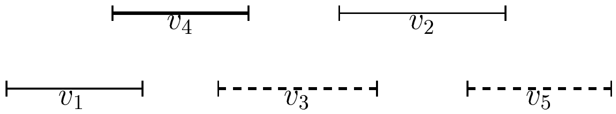
Algorithmus nennen wir jede wohldefinierte Berechnungsvorschrift, die eine Eingabe in eine Ausgabe transformiert. Ein Algorithmus zur Berechnung einer Färbung eines Graphen G etwa transformiert die Eingabe G in eine Partition $(V_1, \dots, V_k)$ der Knoten von G.

$$G \longrightarrow \boxed{\text{Algo.}} \longrightarrow (V_1, \dots, V_k)$$

Ein Algorithmus heißt *korrekt*, wenn er jede Instanz, d.h. jede zulässige Eingabe in eine gültige Ausgabe transformiert. (In unserem Fall müssen dazu alle V_i unabhängige Mengen in G sein). Insbesondere muß der Algorithmus dann auch terminieren.

Beispiel. Wir wollen einen Algorithmus *Coloring* zur Berechnung einer optimalen Färbung eines Intervallgraphen entwickeln.

Da wir als Ausgabe eine Partition der Knoten erwarten, könnte man versuchen, alle Partitionen zu untersuchen, ob sie einer guten Färbung entsprechen, und dann die kleinste davon ausgeben. Dieses Verfahren wäre sicherlich zu aufwendig. Für eine effizientere Lösung erinnern wir uns an den GREEDYCOLORING–Algorithmus, der bei einer vorgegebenen Reihenfolge der Knoten jeden Knoten mit der „kleinsten" verfügbaren Farbe färbt.

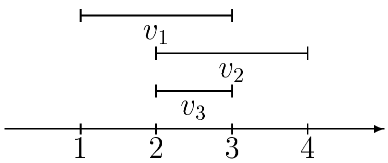


Beobachtung. Um die Intervalle im Bild oben in der Reihenfolge $v_1v_2v_3v_4v_5$ zu färben, benötigt GREEDYCOLORING 3 Farben. Färbt er sie in der Reihenfolge $v_1v_4v_3v_2v_5$, so genügen 2 Farben. Die Reihenfolge der Knoten ist also entscheidend für die Anzahl der Farben. Das Beispiel legt nahe, die Intervalle von „links nach rechts" zu ordnen, was allerdings vorraussetzt, daß eine Intervalldarstellung des Graphen gegeben ist.

Für unseren Algorithmus soll das so realisiert sein, daß die Eingabe aus einem Array A der Größe $2n$ besteht, dessen Einträge Records mit folgenden Feldern sind:

$A[i]$.val	eine reelle Zahl als Intervallgrenze
$A[i]$.ty	± 1 je nachdem, ob Intervallanfang oder -ende
$A[i]$.vert	der Knoten, für den das Intervall steht

Ein Intervallgraph und der den Graphen repräsentierende Array:



i	1	2	3	4	5	6
val	1	3	4	2	2	3
ty	$+1$	-1	-1	$+1$	$+1$	-1
vert	1	3	2	2	3	1

Bevor wir den Algorithmus zum Färben eines Intervallgraphen in Pseudo-Code formulieren treffen wir noch folgende Vereinbarungen:

(1) Sortiert wird nach aufsteigenden val-Werten. Bei gleichen val-Werten sollen Eintäge mit ty $= +1$ vor solchen mit ty $= -1$ kommen. Wenn $I_1 \cap I_2 = \{v\}$ für zwei Intervalle I_1 und I_2 und wenn v Endpunkt von I_1 und Anfangspunkt von I_2 ist, dann denken wir uns I_1 ein wenig nach rechts und I_2 nach links verlängert Wenn nun „ein wenig“ klein genug ist gewählt wurde, dann hat diese Änderung keinen Einfluß auf den Graphen.

(2) Die Verwaltung der zulässigen Farben soll mit einem Stack erfolgen. Wir färben also *nicht* mit der jeweils kleinsten verfügbaren, sondern mit der zuletzt freigewordenen Farbe, da sonst eine aufwendigere Datenstruktur, etwa ein Heap mit Einfügekosten $O(\log(|\text{Farben}|)$, nötig wäre.

(3) Als Farben nehmen wir die Zahlen $1, 2, \ldots$, die Ausgabe des Algorithmus soll die Färbung als knotenindizierter Array C und die höchste verwendete Farbe *maxcol* sein.

```
COLORING(A)
  sort(A)
  stack ← empty
  maxcol ← 0
  for i = 1 to 2n
      if A[i].ty = +1 then
          if stack = empty then
              maxcol++
              s ← maxcol
          else
              s ← pop(stack)
          C[A[i].vert] ← s
      else
          push(C[A[i].vert],stack)
  return C, maxcol
```

Lemma 1. *Coloring ist ein korrekter Algorithmus zur Berechnung von $\chi(G)$ und einer optimalen Färbung für Intervallgraphen.*

Beweis. Zum Beweis, daß die Färbung, die der Algorithmus offensichtlich liefert, optimal ist, zeigen wir per Induktion die Gültigkeit folgender Invariante: Zu jeder Zeit ist der induzierte Subgraph H der k bisher gesehenen Knoten mit $\omega(H)$ Farben gefärbt. Da $\omega(G) \leq \chi(G)$ für jeden Graphen G gilt, genügt das um die Optimalität der Färbung zu beweisen.

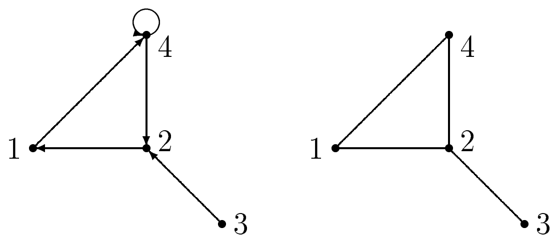
Am Anfang ist der leere Subgraph H mit $\omega(H) = 0$ Farben optimal gefärbt. Sei jetzt $k > 0$. Für $A[k].\mathrm{ty} = -1$ kommt kein neuer Knoten dazu und H ändert sich nicht, es genügt also den Fall $A[k].\mathrm{ty} = +1$ zu betrachten. Ist in diesem Fall der Stack leer, dann gibt es zu jeder der maxcol bisher verwendeten Farben einen Knoten dieser Farbe, dessen Intervallende noch nicht erreicht ist. Diese Knoten bilden also eine maxcol-Clique im induzierten Subgraphen H der $k-1$ schon gesehenen Knoten und sind alle zum Knoten k benachbart. Da nach Induktionsvorraussetzung $\omega(H) = \mathrm{maxcol}$ gilt, folgt $\omega(H+k) = \mathrm{maxcol} + 1$ und die Invariante bleibt gültig. Wenn andrerseits der Stack nicht leer ist, dann ist k mit höchstens $\mathrm{maxcol} - 1$ der bisher gesehenen Knoten adjazent und $\omega(H) = \omega(H+k) = \mathrm{maxcol}$ nach Induktionsvorraussetzung. $\square$

Laufzeitanalyse. Der Stack kann so implementiert werden, daß jede Stackoperation in konstanter Zeit ausführbar ist. Außer der Sortierzeit benötigt der Algorithmus damit in jedem der $2n$ Schleifendurchläufe nur noch konstante Zeit c, also insgesamt weniger als $c2n$. Das ist lineare Zeit $O(n)$. Sind die Intervallgrenzen ganze Zahlen zwischen 0 und $100n$ dann geht auch das Vorsortieren der Intervallgrenzen in $O(n)$. Andernfalls kann der Sortieraufwand $O(n \log n)$ sein.

Übung 1. *Ein Algorithmus zur Berechnung von κ für Intervallgraphen.*

3.2 Datenstrukturen für allgemeine Graphen

Zur Repräsentation von Graphen im Rechner werden im wesentlichen die beiden Datenstrukturen *Adjazenzmatrix* und *Adjazenzlisten* verwendet, die wir am Beispiel der folgenden beiden Graphen illustrieren wollen.

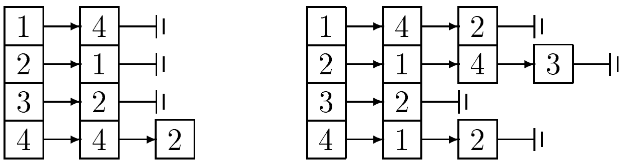


Adjazenzmatrix. In einer mit den Knoten des Graphen indizierten $n \times n$ - Matrix wird die „*ist-Kante*“- Relation mit Einträgen 0 und 1 abgelegt. Im Fall eines ungerichteten Graphen ist die Matrix symmetrisch. Der Speicheraufwand für Adjazenzmatrizen beträgt $O(n^2)$.

0	0	0	1
1	0	0	0
0	1	0	0
0	1	0	1

0	1	0	1
1	0	1	1
0	1	0	0
1	1	0	0

Adjazenzlisten. Für jeden Knoten wird eine Liste seiner Nachbarn angelegt, i.A. ohne bestimmte Ordnung. Für ebene Graphen werden allerdings meist zyklische Adjazenzlisten verwendet, in denen die Nachbarn eines Knoten in der Reihenfolge, in der sie, etwa im Uhrzeigersinn, in der Ebene um ihn herum liegen, angeordnet sind. Der Speicheraufwand für Adjazenzlisten ist $O(n+m)$ bei gerichteten bzw. $O(n+2m)$ bei ungerichteten Graphen.



Der Zeitaufwand für die Grundoperationen im Vergleich ist in der folgenden Tabelle zusammengestellt.

	AdjMat	AdjList	AdjList als bal. Suchbaum
Frage nach Kante ij	$O(1)$	$O(d_i)$	$O(\log(d_i))$
Besuch aller Nachbarn von i	$O(n)$	$O(d_i)$	$O(d_i)$
Besuch aller Kanten	$O(n^2)$	$O(n+m)$	$O(n+m))$
Löschen einer Kante ij	$O(1)$	$O(d_i+d_j)$	$O(\log(d_i)+\log(d_j))$
Einfügen einer Kante ij	$O(1)$	$O(d_i+d_j)$	$O(\log(d_i)+\log(d_j))$

Übung 2. *Eine* Supersenke *in einem orientierten Graphen* $\vec{G} = (V, A)$ *ist ein Knoten* $v \in V$ *mit* $d_v^- = n - 1$*.it Adjazenzmatrizen kann in* $O(|V|)$ *Zeit entschieden werden, ob* $\vec{G}$ *eine Supersenke enthält.*

Übung 3. *Zu einem ungerichteten Graph* $G = (V, E)$ *ist* G^2 *definiert als*

$$G^2 = (V, \{uw |\, \exists v \in V : uv, vw \in E\}).$$

Entwirf und analysiere Algorithmen zur Berechnung von G^2*. (Eingabe: Adjazenzmatrix oder Adjazenzlisten)*

3.3 Depth-First-Search

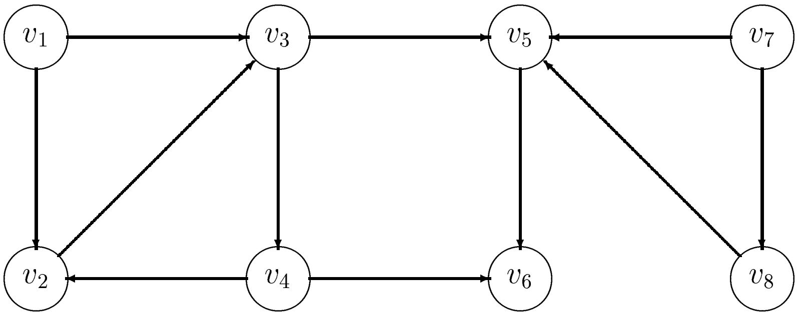
Depth-First-Search, oder Tiefensuche, ist ein Algorithmus zum Durchsuchen eines Graphen, der auf der Idee basiert, von einem Knoten aus rekursiv alle anderen zu besuchen. Der Algorithmus DFS bekommt als Eingabe einen Graphen G, der durch eine Adjazenzmatrix oder Adjazenzlisten gegeben ist. DFS ruft für jeden noch nicht besuchten Knoten v die Prozedur DFSexplore(v) auf, die sich wieder selbst mit den noch nicht besuchten Nachbarn von v rekursiv aufruft, bis alle von v aus erreichbaren Knoten besucht sind. Die im Algorithmus vorgenommene Markierung der Knoten zeigt zu jeder Zeit, ob ein Knoten noch nicht besucht (white), noch in Bearbeitung (grey), oder fertig bearbeitet (black) ist. Als Rückgabewert bekommen wir im Array e-mark eine Partition der Kanten in Baumkanten (dfs-edge), die einen aufspannenden Wald des Graphen bilden, Rückwärtskanten (backedge) und sonstige, sowie in dfs-num die Reihenfolge, in der die Knoten besucht wurden.

```
DFS(G)
  for all v ∈ V
      mark(v) ← white
      num ← 0
  for all v ∈ V
      if mark(v) = white then
          DFSexplore(v)
  return e-mark, dfs-num
```

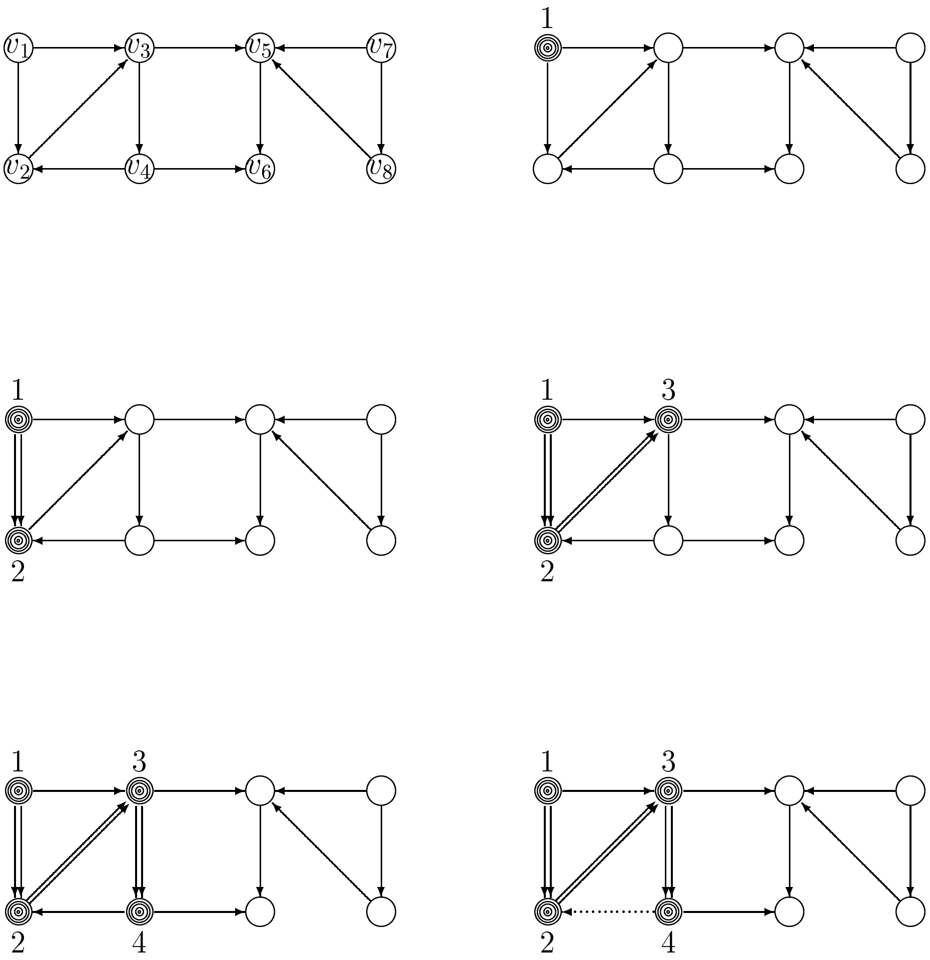
DFSexplore(v)

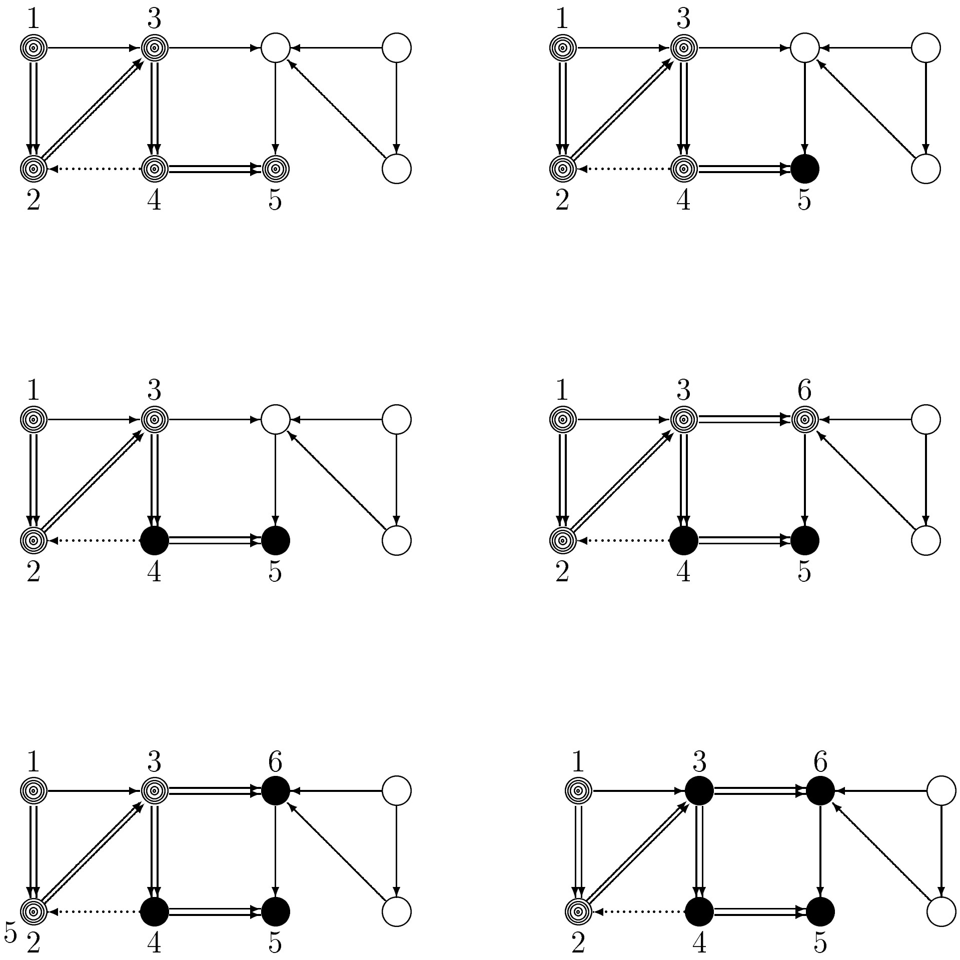
  mark(v) $\leftarrow$ grey
  num++
  dfs-num(v) $\leftarrow$ num
  for all $w \in$ Adj(v)
    if mark(w) = white **then**
      e-mark(vw) $\leftarrow$ dfs-edge
      DFSexplore(w)
    else
      if mark(w) = grey **then**
        e-mark(vw) $\leftarrow$ backedge
  mark(v) $\leftarrow$ black
  return

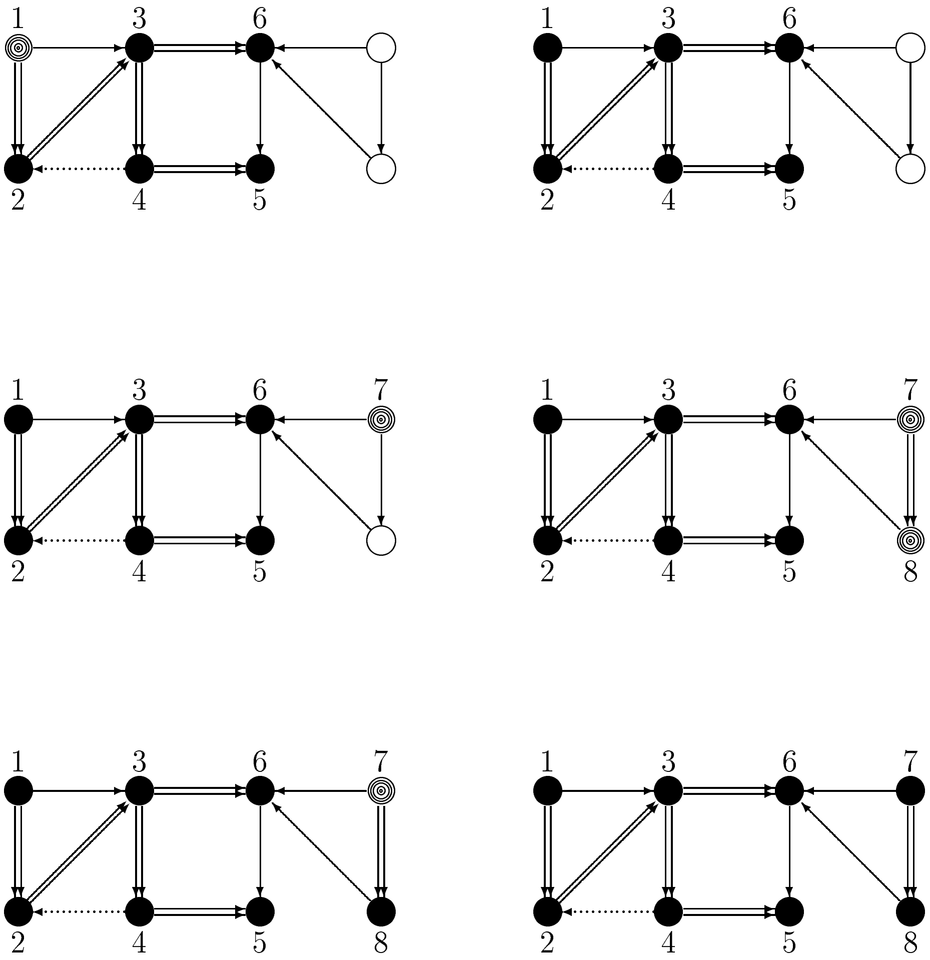
Die Arbeitsweise des Algorithmus und den sukzessiven Aufbau des DFS-Waldes wollen wir uns am Beispiel des folgenden gerichteten Graphen verdeutlichen.



Der Ablauf von DFS ist auf den nächsten 3 Seiten dargestellt. Baumkanten doppelt und Rückwärtskanten gestrichelt gezeichnet und die dfs-num wird an die Knoten geschrieben.


v_1
v_2
v_3
v_4
v_5
v_6
v_7
v_8
1
2
3
4


1 3
2 4 5
1 3
2 4 5
1 3
2 4 5
1 3 6
2 4 5
1 3 6
5 2 4 5
1 3 6
2 4 5



3.4 Eigenschaften von DFS

5. Vorlesung, 7. November 95
Mitschrift: Carola Wenk

1. DFS besucht alle Knoten und alle Kanten:
   Weiße Knoten werden nur in der ersten Zeile von DFSexplore umgefärbt. Die zweite Schleife *for all* $v \in V$ in DFS stellt sicher, daß alle Knoten besucht werden. Alle Kanten werden in der Schleife *for all* $w \in Adj(v)$ in DFSexplore besucht.

2. Laufzeit $\mathcal{O}(|V| + |E|)$:
   Die Initialisierung aller Knoten in DFS benötigt $\mathcal{O}(|V|)$ Zeit. Für jede Kante $v_i v_j$ wird DFSexplore genau einmal, ohne weitere rekursive Aufrufe von DFSexplore, ausgeführt, da nach einem Aufruf beide Knoten grau sind. $\Rightarrow \mathcal{O}(|E|)$.

3. Die Kantenmenge $A_{dfs} := \{e : e\_mark(e) = dfs\_edge\}$ ist ein gerichteter Wurzelwald auf V.

   Beweis.

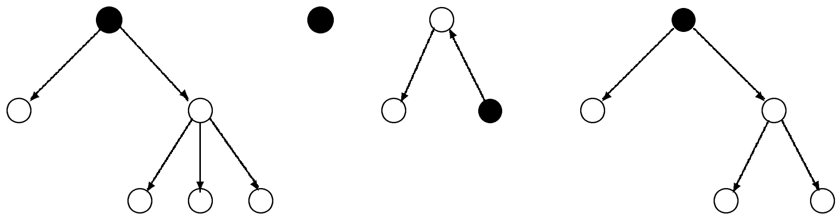
   (a) G ist azyklisch:
   Für $e\_mark(v, w) \leftarrow dfs\_edge$ muß $mark(w) = white$ sein. Anschließend wird $mark(w) = grey$ gesetzt, und später wird $mark(w)$ nie wieder weiß. $\Rightarrow$ Es gibt kein $v' \neq v$ mit $e\_mark(v', w) = dfs\_edge$. $\Rightarrow indeg \leq 1$ für alle Knoten zu allen Zeiten.

   Angenommen, es gibt einen Kreis $v_1, ..., v_k$ in G. O.B.d.A. sei v_k der im Kreis zuletzt besuchte Knoten. Dann gilt $mark(v_i) = grey$ für alle $1 \leq i \leq k-1$. $\Rightarrow e\_mark(v_k, v_1) = backedge$, d.h., die Kante $v_k v_1$ ist keine Kreiskante. Widerspruch.

   (b) Wurzeln sind Knoten mit $indeg = 0$

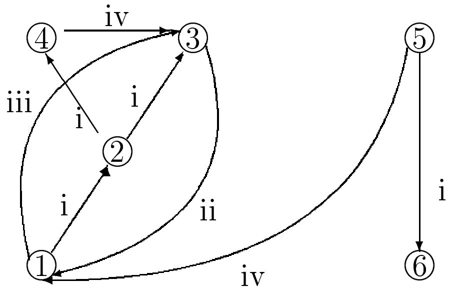
   (c) G ist gerichtet

   □



Beispiel für einen gerichteten Wurzelwald

4. Klassifikation der Kanten:
   (i) Baumkanten
   (ii) Rückwärtskanten
   (iii) Vorwärtskanten
   (iv) Kreuzende Kanten



Bemerkung.

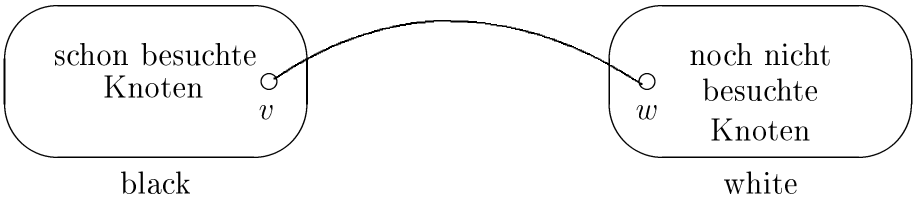
- Die einzigen Kanten, die zwei Komponenten eines DFS-Waldes eines gerichteten Graphen miteinander verbinden, sind *kreuzende Kanten.*
- Ist G ein ungerichteter Graph, so gibt es keine Kanten die von einem Zweig des Wurzelwaldes zu einem anderen zeigen. Also keine kreuzenden kanten.

Übung 4. *Ein Digraph ist azyklisch* $\Longleftrightarrow$ *Es gibt keine Rückwärtskanten.*

Hinweis: ($\Leftarrow$) wird indirekt bewiesen. Man betrachte den Knoten mit niedrigster *dfs-num* auf dem Kreis.

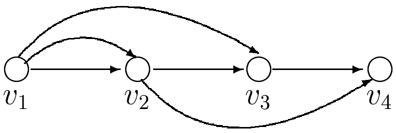
Lemma 2. *Ein (ungerichteter) Graph $G = (V, E)$ ist zusammenhängend $\Longleftrightarrow$ Der DFS-Wald von G ist ein Baum.*

Beweis. ($\Leftarrow$) DFS-Wald $\subseteq E$
($\Rightarrow$) Angenommen, es gibt $\geq$ 2 Komponenten. Betrachte den Zustand des Algorithmus in dem Moment, in dem die 2. Wurzel erzeugt wird:



Die Kante vw existiert, da G als zusammenhängend vorausgesetzt war. Der Aufruf *DFSexplore*(v) hätte die Kante vw besuchen müssen. Widerspruch. □

Lemma 3. *Sei $D = (V, A)$ ein DAG. Die Reihung der Kinder nach absteigender Schwarzfärbezeit definiert eine topologische Sortierung von D.*



Beispiel einer topologischen Sortierung. (Alle Kanten gehen von links nach rechts.)

Beweis. Sei $(u, v) \in A$. Wir wollen zeigen, daß u nach v schwarz gefärbt worden ist.

Wir betrachten den Moment, in dem die Kante (u, v) untersucht wurde. Der Knoten u ist dann grau gefärbt, und v kann folgende verschiedene Zustände haben:

- *schwarz:* $\surd$
- *grau:* Kante (u, v) ist backedge $\overset{\text{Übung}}{\Longrightarrow}$ D ist nicht azyklisch.
- *weiß:* Es wird *DFSexplore* aufgerufen und erst zu u zurückgekehrt, wenn v schwarz ist. □

Wir haben gerichtete azyklische Graphen, sogenannte Dags kennengelernt. Die Höhenfunktion h eines Dag ist rekursiv definiert als.

$$h(v) = \begin{cases} 1 & \text{wenn } v \text{ Quelle (also } indeg(v) = 0) \\ 1 + \max\limits_{(w,v) \in A} h(w) & \text{sonst} \end{cases}$$

Übung 5. *Benutze einen Aufruf von DFS, um die Höhenfunktion eines Dag zu berechnen.*

Bemerkung. In Kapitel 2 hatten wir minimale Antikettenzerlegung $A_1, \ldots, A_k$ einer Ordnung in Verbindung mit optimalen Färbungen des Vergleichbarkeitsgraphen gebracht. Da die Abbildung $h : x \mapsto i$ wenn $x \in A_i$ gerade die Höhenfunktion der Ordnung ist, folgern wir, daß Vergleichbarkeitsgraphen (wenn die transitive Orientierung gegeben ist) in linearer Zeit optimal gefärbt werden können.

Zum Kapitel

Eine umfassende Einführung in Datenstrukturen und Algorithmen ist das Buch von Cormen, Leiserson und Rivest [10], das mir auch als Quelle für dieses Kapitel gedient hat.

4 Problemklassen und Komplexität

Beispiel. Wie aufwendig ist es, das folgende Problem zu lösen?

Problem **Π**:	Graphen Färben
Instanz:	Ungerichteter Graph G
Frage:	Wie groß ist $\chi(G)$?

Natürlich können wir nicht erwarten ein Problem auf einem ‘großen’ Graphen genauso schnell lösen zu können wie auf einem ‘kleinen’. Bei Komplexitätsbetrachtungen berücksichtigt man daher die Problemgröße die man durch die *Kodierungslänge* $|\Pi|$ mißt. Bei Graphenproblemen wird die Kodierungslänge meistens als $n + m$ angenommen. Das entspricht der Darstellung des Graphen durch Adjazenzlisten.

Definition. Eine *obere Schranke* für Πist eine Funktion f und ein Algorithmus, der Π in $\mathcal{O}(f(|\Pi|))$ Schritten löst. Eine *untere Schranke* ist eine Funktion g, so daß jeder Algorithmus, der Π löst, zumindest $\Omega(g(|\Pi|))$ Schritte benötigt.

Übung 1. *Gib obere und untere Schranken für das Problem "Graphen Färben" an.*

Definition. Wenn f durch ein Polynom in $|\Pi|$ dominiert werden kann, nennt man das Problem *effizient lösbar*. Die Klasse der effizient lösbaren Probleme nennt man $\mathcal{P}$.

4.1 Die Problemklasse $\mathcal{NP}$

$\mathcal{NP}$ ("*n*ichtdeterministisch *p*olynomiell") enthält solche Probleme, die 'mit Einsagen' in polynomieller Zeit gelöst werden können.

Definition. Ein *deterministischer Prozeß* geht in jedem Schritt von einem Zustand in einen anderen über. Mithilfe einer *nichtdeterministischen Erweiterung* kann ein Prozeß verschiedene Kopien seiner selbst im nächsten Zustand starten.

Der Begriff einsagen in obiger informeller Definition von $\mathcal{NP}$ meint, daß der Einsagende dem Algorithmus mitteilt welche der Kopien mit Aussicht auf Erfolg zu verfolgen ist.

Definition. Ein *Entscheidungsproblem* ist ein Problem dessen Lösung in der Gestalt einer Ja/Nein Antwort gegeben werden kann.

Beispiel. Entscheidungsvariante von "Graphen Färben":

Instanz: (G, k)
Frage: $\chi(G) \leq k$?

Im folgenden werden wir uns auf die Diskussion von Entscheidungsproblemen beschränken. Wir Formalisieren den Begriff des nichtdeterministischen Algorithmus für Entscheidungsprobleme, indem wir Algorithmen für solche Probleme spezielle Operationen erlauben:

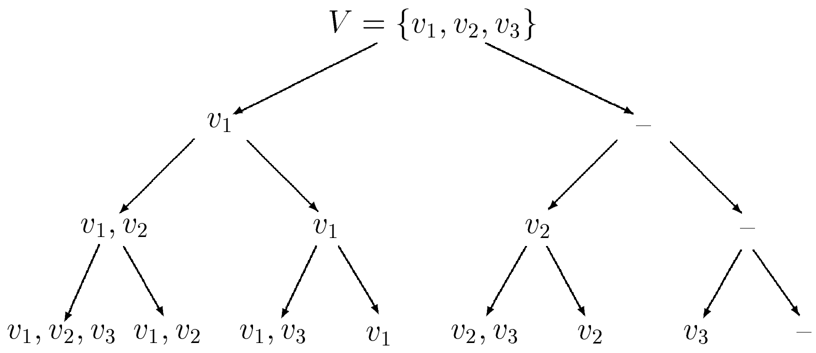
$x \leftarrow \text{choice}(S)$	Für jedes $y \in S$ wird eine Kopie mit $x \leftarrow y$ gestartet.
failure	Bricht die aktuelle Kopie ab. Wenn die aktuelle die letzte Kopie war gibt der Algorithmus "Nein" aus.
succes	Bricht alle Kopien ab. Der Algorithmus gibt "Ja" aus.

Die Laufzeit eines nichtdeterministischen Algorithmus ist definiert als das Maximum der Laufzeiten die für die einzelnen Kopien aufgewandt wurden.

Beispiel. Das Problem: Enthält G eine k-Clique? läßt sich durch den folgenden nichtdeterministischen Algorithmus in polynomieller Zeit lösen.

```
Clique(G, k)
  A ← ∅
  for all v ∈ V
      A ← choice({A ∪ {v}, A})
  if |A| < k then
      failure
  for all v ≠ w ∈ A
      if vw ∉ E then
          failure
  success
```

Der abgebildete Baum zeigt wie der Algorithmus durch die erste Schleife für jede Teilmengen A von V eine Kopie startet. In jedem Blatt wird dann untersucht $|A| \geq k$ und fall ja, ob das gegebene A eine Clique ist.



Damit ist gezeigt: Clique $\in \mathcal{NP}$.

Klarerweise ist $\mathcal{P} \subseteq \mathcal{NP}$. Das große offene Problem der Komplexitätstheorie ist ob $\mathcal{P}$ und $\mathcal{NP}$ tatsächlich verschieden sind.

Vermutung. $\mathcal{P} \neq \mathcal{NP}$.

Bemerkung. Unsere Definition von nichtdeterministischen Algorithmen ist nicht Symmetrisch. Vertauscht man die Rollen von success und failure so, daß success eine und failure alle Kopien abbricht dann sind die zu $co - \mathcal{NP}$ gehörenden Probleme die polynomiell Lösbaren.

Vermutung. $co - \mathcal{NP} \neq \mathcal{NP}$.

Übung 2. *Zeige:* $co - \mathcal{NP} \neq \mathcal{NP} \Longrightarrow \mathcal{P} \neq \mathcal{NP}$.

4.2 Polynomielle Transformationen

6. Vorlesung, 14.11.95
Mitschrift: Christof Schultz

Definition. Seien Π_1 , Π_2 Entscheidungsprobleme. Dann nennt man

$$F : \Pi_1 \longrightarrow \Pi_2$$

eine *polynomielle Transformation* wenn gilt:

(1) F ist polynomiell berechenbar.

(2) JA-Instanzen von Π_1 werden auf JA-Instanzen von Π_2 abgebildet.

(3) NEIN-Instanzen von Π_1 werden auf NEIN-Instanzen von Π_2 abgebildet.

Wenn Π_1 polynomiell aus Π_2 berechnet werden kann schreibt man $(\Pi_1 \propto \Pi_2)$.

Bemerkung. Aus $\Pi_1 \propto \Pi_2$ und $\Pi_2 \in \mathcal{P}$ folgt : $\Pi_1 \in \mathcal{P}$.

Beweis. Die Summe von polynomiell beschränkten Funktionen ist polynomiell beschränkten. □

4.3 NP-vollständigkeit und das SAT Problem

Definition. Ein Problem Π heißt *NP-hart*, wenn $\Pi' \propto \Pi$ für alle $\Pi' \in \mathcal{NP}$. Ein Problem Π ist *NP-vollständig*, wenn $\Pi \in \mathcal{NP}$ und Π NP-hart.

Lemma 1. *Wenn* $\Pi_1 \propto \Pi_2$ *und* $\Pi_1, \Pi_2 \in \mathcal{NP}$, *dann folgt aus* Π_1 *NP-vollständig auch* Π_2 *NP-vollständig.*

Beweis. Man muß für alle $\Pi' \in \mathcal{NP}$ zeigen, daß $\Pi' \propto \Pi_2$ gilt. Betrachten wir nun $\Pi' \overset{F_1}{\propto} \Pi_1 \overset{F_2}{\propto} \Pi_2$, so gilt auch $\Pi' \overset{F_1 \circ F_2}{\propto} \Pi_2$. □

Wenn von einem NP-vollständigen Problem gezeigt werden kann, daß es polynomiell lösbar ist, dann folgt sofort $\mathcal{P} = \mathcal{NP}$. In diesem Sinn sind die NP-vollständigen Probleme die schwierigsten Probleme in NP. Vorerst ist gar nicht klar, daß es überhaupt NP-vollständige Probleme gibt. Wenn jedoch von einem Problem Π gezeigt ist, daß es NP-vollständig ist, dann gibt uns das Lemma eine Methode an die Hand um diese Eigenschaft für weitere Probleme zu beweisen.

Historisch wurde der erste NP-Vollständigkeitsbeweis von Cook 1971 geführt. Er bewies NP-vollständigkeit des Problems *Satisfiability* (kurz *SAT*) das wir nun kennenlernen wollen.

Defintion. Ein *Literal* in einer boolschen Formel, ist das Vorkommen einer Variabel oder ihre Negation. Eine boolsche Formel ist eine *CNF-Formel* (conjunctive normal form), wenn sie aus Clausen besteht, die durch logische UND verknüpft sind. Jede Clause besteht dabei aus Literalen, die jeweils durch logische ODER miteinander verknüpft sind.

Beispiel für eine CNF-Formel:

$$(x_1 \vee x_3 \vee x_5) \wedge (\overline{x_2} \vee \overline{x_3} \vee x_4) \wedge (\overline{x_2} \vee \overline{x_5}) \wedge \underbrace{(x_1 \vee \overline{x_2} \vee x_3 \vee \overline{x_4})}_{Clause}$$

Hierbei sind x_i Variabeln und $x_i, \overline{x_i}$ Literale. Für $x_1 = w$ und $x_2 = f$ ist der gesamte Ausdruck wahr. So etwas nennt man eine *Wahrheitsbelegung*.

SAT Problem: Gegeben: Eine CNF-Formel F. Frage: Gibt es eine Belegung der Variabeln x_i mit $x_i \in \{w, f\}$ die F wahr macht?

Satz 2. *SAT ist NP-vollständig.*

4.4 Das Vertex Cover Problem

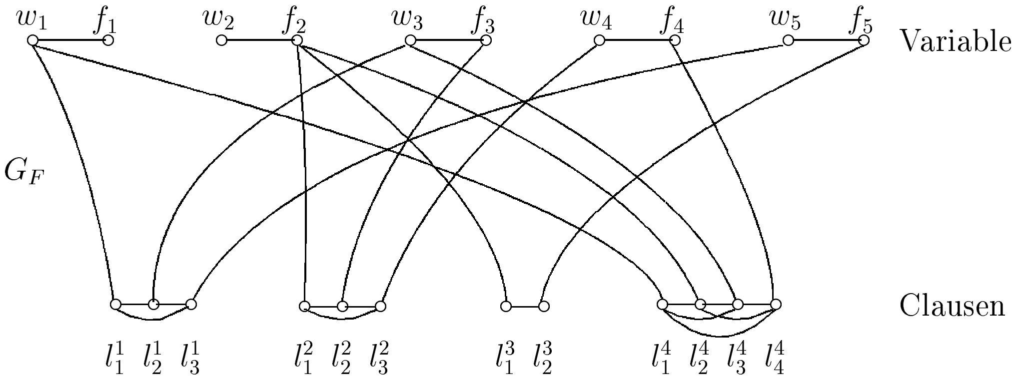
Eine *Knotenüberdeckung* (engl. *vertex cover*) in $G = (V, E)$ ist eine Knotenmenge $W \subseteq V$, so daß für alle $e \in E$ gilt: $e \cap W \neq \emptyset$, das heißt W berührt jede Kante von G.

Vertex Cover Problem:
Gegeben: ein Graph $G = (V, E)$ und eine natürliche Zahl $k \in \mathbb{N}$.
Frage: Gibt es eine Knotenüberdeckung W der Größe $|W| \leq k$ für G?

Beispiel.

Lemma 3. *SAT* $\propto$ *Vertex Cover*

Beweis. Sei $F = C_1 \wedge C_2 \wedge \ldots \wedge C_r$ eine gegebene CNF-Formel mit den Clausen $C_i = (l_1^i \vee l_2^i \vee \ldots \vee l_{a_i}^i)$ und den Literalen $l_j^i \in \{x_1, \ldots, x_n, \overline{x_1}, \ldots \overline{x_n}\}$, wobei $x_1, \ldots, x_n$ die Variabeln sind. Zu jeder Clause C_i definieren wir eine Clique mit den Knoten $l_j^i, j = 1 \ldots a_i$. Zu jeder Variable definieren wir eine Kante $\{w_i, f_i\}$. Nun verbindet man noch l_i^j mit w_a, wenn $l_j^i = x_a$, andernfalls, d.h. wenn $l_j^i = \overline{x_a}$ verbindet man l_i^j mit f_a. Der durch diese Konstruktion enstandene Graph G_F zu der CNF-Formel aus dem Beispiel sieht wie folgt aus:



Beobachtung. Für jede Variable ist ein Knoten und für jede Clause mit a Literalen sind $a-1$ Knoten in jedem Vertex Cover nötig. Insgesamt also zumindest $n + \sum_{i=1}^{r}(a_i - 1)$ Knoten.

Behauptung. Es gibt ein Vertex Cover mit $n + \sum_{i=1}^{r}(a_i - 1)$ Knoten in G_F genau dann wenn F erfüllbar ist.

($\Leftarrow$) Sei $\Gamma : x_i \to \Gamma(x_i) \in \{w, f\}$ eine erfüllende Belegung von F. Sei $W \subseteq V$ wie folgt definiert:

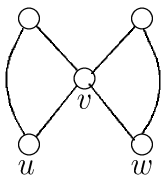
(1) W enthält w_i, wenn $\Gamma(x_i) = w$ und f_i, wenn $\Gamma(x_i) = f$

(2) $\{l_1^i, \ldots, l_{b_{i-1}}^i, l_{b_{i+1}}^i, \ldots, l_{a_i}^i\} \subset W$ wenn $l_{b_i}^i$ das erste wahre Literal in C_i ist.

Behauptung. W ist eine Knotenüberdeckung.
Sei e eine Kante in G_F, dann ist zu zeigen, daß einer der Endpunkte von e in W ist. Angenommen es existiert eine Kante e die nicht überdeckt ist, dann gilt entweder $e = (l^i_j, w_s)$ mit $l^i_j = x_s$ oder $e = (l^i_j, f_s)$ mit $l^i_j = \overline{x_s}$. Damit aber $l^i_j \notin W$ muß $j = b_i$ sein. Daraus folgt, daß l^i_j ein wahres Literal ist. Das wiederum bedeutet, daß $\Gamma(x_s) = w_s$ oder $\Gamma(x_s) = f_s$. Daraus folgt, daß $w_s \in W$, bzw. $f_s \in W$ ist. Dies steht im Widerspruch dazu, daß die Kante e nicht überdeckt wird. Also ist W ein Vertex Cover mit $n+\sum_{i=1}^{r}(a_i-1)$ Knoten.
($\Rightarrow$) Sei W ein Vertex Cover mit $|W| = n + \sum_{i=1}^{r}(a_i - 1)$. Dann enthält W genau einen der beiden Knoten (w_i, f_i). Wähle diesen Knoten zur Definition von Γ. Nun muß man zeigen, daß das eine erfüllende Belegung ist, d.h. in jeder Clause ist ein wahres Literal. Das gilt, weil in jeder Clause ein Knoten nicht in W liegt. $\square$

Lemma 4. *$G = (V, E)$ ein Graph. $W \subseteq V$ ist ein minimum Vertex Cover in G genau dann wenn $\overline{W} = V - W$ eine maximum unabhängige Menge in G ist.*

Beispiel.



Hier ist v eine maximale (nicht vergrößerbare) unabhängige Menge, $\{u, w\}$ hingegen ist eine maximum unabhängige Menge, d.h. eine von maximaler Kardinalität und $V \setminus \{u, w\}$ ein minimum Vertex Cover.

Beweis. Es gilt : W Vertex Cover $\iff$ $\overline{W}$ unabhängige Menge.
Dann folgt: G enthält Vertex Cover der Größe $\leq k \Leftrightarrow G$ enthält unabhängige Menge der Größe $\geq n - k$. Damit sind die Probleme minimum Vertex Cover und maximum unabhängige Menge äquivalent. $\square$

Folgerung 1: Vertex Cover $\propto$ unabhaängige Menge, d.h. unabhängige Menge ist NP-vollständig. Nehme als Transformation $(G \to G; k \to n - k)$

Folgerung 2: Unabhängige Menge $\propto$ Clique, d.h Clique ist NP-vollständig. Nehme als Transformation $(G \to \overline{G}; k \to k)$

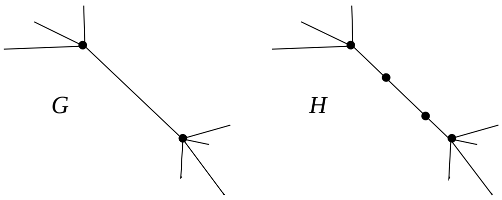
Bemerkung. Clique bezeichnet das Entscheidungsproblem mit Eingabe (G, k) und der Frage: Enthält Graph G eine k-Clique? Betrachten wir zum Vergleich das Problem Clique$_k$ mit Eingabe G und der Frage: Enthält Graph G eine k-Clique? Der vermeintlich kleine Unterschied, im ersten Problem ist k variabel, im zweite konstant, ist doch wesentlich: Während Clique wie eben gezeigt NP-vollständig ist, kann das Problem Clique$_k$ in polynomieller Zeit $O(n^k)$ gelöst werden.

7. Vorlesung, 16.11.95
Mitschrift: Susanna Gebauer

Lemma 5. *Independent Set läßt sich reduzieren auf Independent Set für dreiecksfreie Graphen:*

$$\textit{Independent Set} \propto \textit{Indep. Set f. dreiecksfr. Graphen.}$$

Beweis. Sei $G = (V, E)$ mit $n = |V|$ und $m = |E|$ gegeben, wir konstruieren $H = (V', E')$ indem wir jede Kante $e \in E(G)$ zweifach unterteilen. Offensichtlich ist H dreiecksfrei.



Behauptung: $\alpha(H) = \alpha(G) + m$.

Sei I unabhängig in G. Man kann I zu einem I' unabhängig in H erweitern, das aus jeder Kante noch wenigstens einen der Unterteilungsknoten enthält. $\Rightarrow \alpha(H) \geq \alpha(G) + m$.

Sei nun I' maximum unabhängige Menge in H. Wir zeigen zuerst, daß wir annehmen dürfen, daß I' einen der beiden Kantenunterteilungsknoten aus jeder Kante $e \in E$ enthält. Seien u, v die Endknoten der Kante und a, b die Unterteilungsknoten, angenommen $a, b \notin I'$

- $u \notin I'$ oder $v \notin I'$ $\Rightarrow$ I' kann vergrößert werden $\Rightarrow$ Widerspruch.
- $u \in I'$. Tausche u und a aus. dadurch bleibt die größe der unabhängigen menge gleich und die Anzahl der Kantenunterteilunsknoten in I' wird vergrößert. Durch Wiederholung erreicht man $|I'| = m + |I' \cap V|$.

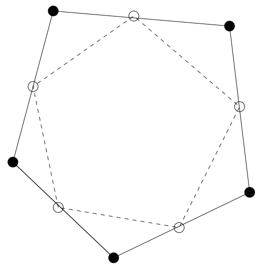
Aus $\alpha(H) = |I'|$ und $|I' \cap V| \leq \alpha(G)$ folgt nun $\alpha(H) \leq m + \alpha(G) \Rightarrow \alpha(H) = m + \alpha(G)$. $\triangle$

Definition. Der *Line-Graph* von $G = (V, E)$ ist der Graph

$$\mathcal{L}(G) = (E, \{(e, e') : e', e \in E, e \cap e' \neq \emptyset\})$$

d.h. e und e' sind benachbarte Knoten in $\mathcal{L}(G) \iff e$ und e' haben einen gemeinsamen Knoten in G.

Beispiel. C_k Kreis, Knoten sind Kanten von $\mathcal{L}(C_k)$.



Lemma 6. *Independent Set läßt sich reduzieren auf Chromatic Number:*

$$\textit{Independent Set} \propto \textit{Chromatic Number}$$

Beweis. Sei $G = (V, E)$ und $H = (V', E')$ wie im vorigen Lemma. Mit K bezeichnen wir den Graphen $\overline{\mathcal{L}(H)}$. Es gibt eine Bijektion (symbolisch $\sim_{bij}$) zwischen Färbungen von K und Cliquenzerlegungen von $\mathcal{L}(H)$.

H dreiecksfrei $\Rightarrow$ Cliquen von $\mathcal{L}(H)$ entsprechen Kantenmengen in H, die einen gemeinsamen Knoten haben (Sternen) $\Rightarrow$ Cliquenzerlegungen von $\mathcal{L}(H) \sim_{bij}$ Knotenüberdeckungen in $H \sim_{bij}$ unabhängige Mengen in H.

Sei $n = |V(G)|$ und $m = |E(G)|$. Benutzt man die Gleichung aus dem Beweis von Lemma 5 so erhält man:

$$\begin{aligned} \chi(K) &= \kappa(\mathcal{L}(H)) = vc(H) = |V(H)| - \alpha(H) \\ &= n + 2m - (\alpha(G) + |E(G)|) = n + m - \alpha(G) \end{aligned}$$

Polynomielle Berechenbarkeit von $\chi(K)$ würde also polynomielle Berechenbarkeit von $\alpha(G)$ implizieren. $\triangle$

Zum Kapitel

Die Standardreferenz für NP-Vollständigkeit ist Garey und Johnson [17]. Aber auch [10] enthält den Beweis des Satzes von Cook und etliche Reduktionsbeweise. Mit unbekannt aber auf deutsch: Reischuck, Einführung in die Komplexitätstheorie, Teubner 1990.

Die hier vorgestellte Definition der Klasse $\mathcal{NP}$ ist eher ungewöhnlich (vergl. mit obigen Büchern). Ich habe sie in Golumbic [18] gefunden. Leider konnte ich den Ansatzes mit den angegebenen Referenzen nicht bis auf seinen Ursprung zurückverfolgen.

5 Einige Klassen χ-perfekter Graphen

Zur Erinnerung:

$\omega(G[W]) = \chi(G[W])$ für alle $W \subseteq V$, dann heißt G χ-perfekt.

$\alpha(G[W]) = \kappa(G[W])$ für alle $W \subseteq V$, dann heißt G α-perfekt.

G ist χ-perfekt genau wenn $\overline{G}$ α-perfekt ist.

5.1 Bipartite Graphen

G sei ein bipartiter Graph. Weil bipartit eine hereditäre Grapheneigenschaft ist, folgt χ-perfekt aus $\omega(G) = \chi(G)$ für jeden bipartiten Graphen.

Lemma 1. *Bipartite Graphen sind χ-perfekt.*

Beweis. Es gibt keine Dreiecke, also ist $\omega(G) = 1$ oder $\omega(G) = 2$. Besteht G nur aus isolierten Knoten, dann ist $\chi = \omega = 1$. Vollständige bipartite Graphen haben $\chi = 2$, ihre Subgraphen $\chi \leq 2$. Enthält G eine Kante, dann gilt also $\chi = \omega = 2$. △

5.2 Komplemente von bipartiten Graphen

Sei $H = \overline{G}$ das Komplement eines bipartiten Graphen $G = (V, E)$. Eine Menge C ist unabhängig in H genau wenn sie Clique in G ist. Also sind die Partitionen von H in unabhängige Mengen genau die Cliquenüberdeckungen von G. Die Cliquen in G sind entweder Kanten oder Knoten. Eine minimale Cliquenüberdeckungen von G erhält man daher indem man eine maximale Anzahl von paarweise disjunkten Kanten mit 1-elementigen Knotenklassen zu einer Überdeckung erweitert. Teilmengen $M \subseteq E$ heissen Matching. Die auf diese Art durch ein Matching M induzierte Cliquenüberdeckung hat Größe $n - |M|$.

Daher gilt:

$$\chi(H) = \kappa(G) = n - \max_{M \text{Matching}} |M|.$$

Offensichtlich gilt auch

$$\omega(H) = \alpha(G) = n - \min_{A \begin{subarray}{c}\text{Vertex}\\\text{Cover}\end{subarray}} |A|.$$

Damit folgt die χ-perfektheit von H aus dem nächsten Satz.

Satz 2. [König-Egervàry] *G bipartit $\Rightarrow$ maximale Größe eines Matchings in G = minimale Größe eines Vertex Covers in G.*

Beweis. Es gilt $|M| \leq |U|$ für jedes Vertex Cover U und jedes Matching M. Sei $\tau(G) = \min\{|U| : U \text{ Vertex Cover in } G\}$. Entferne so viele Kanten wie möglich aus G ohne $\tau(G)$ zu ändern. Für den so erzeugten Subgraphen G' von G gilt also (1) $\tau(G') = \tau(G)$ und (2) $\forall e \in E(G')$ ist $\tau(G' - e) < \tau(G')$.
Behauptung: G' ist ein Matching.
Andernfalls finden wir Kanten e und e' die beide einen Knoten v enthalten in G'. Für $e^* \in \{e, e'\}$ wähle im Graphen $G' - e^*$ ein Vertex Cover U_{e^*} der Größe $\tau(G') - 1$. Es gilt: $e^* \cap U_{e^*} = \emptyset$, sonst wäre U_{e^*} ein Vertex Cover der Größe $\tau(G') - 1$ von G'.

Betrachte $G'' = G'[v + (U_e \triangle U_{e'})]$. Ist $t = |U_e \cap U_{e'}|$, dann ist $|V(G'')|$ genau $1 + 2(\tau(G) - 1) - 2t = 2(\tau(G) - t - 1) + 1$. G'' ist zweifärbbar, weil G'' Subgraph des bipartiten Graphen G ist. Sei T die kleinere Farbklasse in einer Zweifärbung von G'', dann ist $|T| \leq \tau(G) - t - 1$.

Behauptung: $S = T + (U_e \cap U_{e'})$ ist Vertex Cover für G'.
Sei f Kante von G', $f \neq e, e'$, dann wird f wird von U_e und von $U_{e'}$ bedeckt:

- Enthalten U_e und U_e' das gleiche Ende von f, dann ist dieses Ende in $U_e \cap U_{e'}$ und also in S.
- $f = (u, u')$ mit $u \in U_e \setminus U_{e'}$ und $u' \in U_{e'} \setminus U_e$. Dann gilt $f \in G''$ und entweder u oder u' ist in T, also in S.

Da $|U_e \cap U_{e'}| = t$ und $|T| \leq \tau(G) - t - 1$ erhalten wir den Widerspruch $|S| \leq \tau(G) - 1$. Also ist G' ein Matching der Größe $\tau(G)$. $\square$

5.3 Line Graphen von bipartiten Graphen

8. Vorlesung, 21. November 95
Mitschrift: Clemens Hendler

Aus der Definition der *Line-Graphen* folgt unmittelbar folgendes

Lemma 3. *Sei G bipartiter Graph, $\mathcal{L}(G)$ der zugehörige* Line-Graph*, dann gelten folgende Entsprechungen:*

$$\begin{array}{rcl} \mathcal{L}(G) & \leftrightarrow & G \\ \textit{Clique} & \leftrightarrow & \textit{Sterne} \\ \textit{unabhängige Menge} & \leftrightarrow & \textit{Matching} \\ \textit{Cliquenzerlegung} & \leftrightarrow & \textit{Vertex Cover} \\ \textit{Färbung} & \leftrightarrow & \textit{Zerlegung der Kanten von } G \textit{ in Matchings} \end{array}$$

Zunächst wollen wir ein wichtiges Werkzeug der Matching-Theorie kennenlernen:

Definition. Ein Pfad $W = v_0, .., v_k$ heißt alternierend (bezüglich eines Matchings M), falls gilt: $(v_{2i}, v_{2i+1}) \notin M$ und $(v_{2i+1}, v_{2i+2}) \in M$.

Beobachtung.

(*i*) Die *nicht*-Matchingkanten entlang eines einfachen alternierenden Pfades bilden ein Matching.

(*ii*) Ist $W := v_0, ..., v_k$ ein alternierender Pfad bzgl. M gerader Länge (k gerade) und ist v_0 kein Matchingknoten, so bilden die Nicht-Matchingkanten entlang W vereinigt mit dem Restmatching wieder ein Matching M', d.h. M ist die symmetrische Differenz von M und W. Es gilt: $V(M') = (V(M) - v_k) \cup v_0$, also $|M'| = |M|$.

(*iii*) Hat W ungerade Länge und sind weder v_0 noch v_k Matchingknoten, dann kann man auf gleiche Weise ein Matching M' konstruieren. In diesem Fall gilt jedoch: $V(M') = V(M) \cup \{v_0, v_k\}$, also $|M'| = |M| + 1$.

Satz 4. [König] *Sei G bipartiter Graph. Dann gilt: Kanten von G können in $\Delta(G)$ Matchings zerlegt werden.*

Beweis. Es genügt zu zeigen: $\exists$ Matching M in G, das alle Knoten vom Grad Δ abdeckt (denn $\Delta(G - M) = \Delta(G) - 1$).

Sei M ein maximum-Matching, das eine maximale Anzahl von Knoten vom Grad Δ abdeckt.

Annahme: ex Knoten x_0 mit $d_{x_0} = \Delta$, der nicht von M abgedeckt wird.

Es gilt: jeder in x_0 startende alternierende Pfad gerader Länge endet in einem Knoten x_{2k} mit $d_{x_{2k}} = \Delta$. Denn nach Beobachtung (ii) gibt es ein weiteres maximum-Matching M', bezüglich dessen x_0 Matchingknoten und x_{2k} nicht Matchingknoten ist. Hätte x_{2k} einen Grad kleiner Δ, so würde M' mit x_0 einen Knoten vom Grad Δ mehr abdecken als M.

Betrachte nun die von allen in x_0 startenden alternierenden Pfaden aufgespannte Knotenmenge V' und den Graphen $G' = G[V']$. Natürlich ist G' wieder bipartit. $X' \subset V'$ bezeichne die Bipartitionsklasse von G', die x_0 enthält, also die Knoten mit geradem Pfadindex und Y' bezeichne die Knotenmenge mit ungeradem Pfadindex. Für $x \in X'$ gilt wie oben gezeigt: $d_x = \Delta$.

Behauptung. $x \in X' \Rightarrow d_x = d'_x$.
Sei $x \in X', xy \in E$, wir müssen zeigen $y \in V'$. Betrachte alt. Pfad W mit Startpunkt x_0 und Endpunkt x. Für $x = x_0$ ist die Beh. klar, da dann xy ein alt. Pfad ist, und somit $y \in V'$. Für x mit Pfadindex $2p+2$ ist die letzte Kante von W eine Matchingkante. Fall 1: $xy \in M$. Dann gilt $y = x_{2p+1} \in V'$. Fall 2: $xy \notin M$. Wenn $y \in W$ dann ist klarerweise $y \in V'$, Andernfalls kann W kann durch die Kante xy zu einem alternierenden Weg der y enthält fortgesetzt werden, also $y \in V'$.

Also gilt: $x \in X' \Rightarrow d'_x = \Delta$, und damit:

$$|Y'|\Delta \geq \sum_{y \in Y'} d'_y = \sum_{x \in X'} d'_x = |X'|\Delta, \qquad \text{also } |Y'| \geq |X'| \tag{$\star$}$$

.

Behauptung. $y \in Y' \Rightarrow y$ liegt auf Matchingkante.
$y \in Y' \Rightarrow$ ex alt. Pfad W mit Startpunkt x_0 und Endpunkt y. Annahme: y kein Matchingknoten. Nach Beob. (iii) ex M' mit $|M'| = |M| + 1$ — Widerspruch zu M ist maximum Matching.

Y' besteht also nur aus Matchingknoten, und für jedes $y \in Y'$ liegt sein Matchingpartner in X'. Außerdem liegt auch x_0 in X', also ein nicht-Matchingknoten, der keinen Matchingpartner in Y' hat, woraus man schließen kann: $|X'| > |Y'|$ — Widerspruch zu $(\star)$. □

Folgerung. G bipartit $\Longrightarrow \mathcal{L}(G)$ ist χ-perfekt.

Beweis. Es gilt: fa. induzierten Subgraphen $S \subseteq \mathcal{L}(G)$ ex ein bipartiter Graph $H \subseteq G$ sod. $\mathcal{L}(H) = S$. (Betrachte dazu $G = (V, E)$, dann $\mathcal{L}(G) = (E, E')$;

sei $V(S) = F \subseteq E$, dann ist $S = \mathcal{L}(G)[F] = (F, E' \cap F \times F) = \mathcal{L}(V, F))$. Mit anderen Worten: diese Klasse ist hereditär.

Also reicht es zu zeigen: $\chi(\mathcal{L}(G)) = \omega(\mathcal{L}(G))$.
$\chi(\mathcal{L}(G))$ entspricht der minimalen Anzahl Matchings, die nötig sind, um $E(G)$ in Matchings zu zerlegen. $\omega(\mathcal{L}(G)) = \Delta(G)$. Satz von König: $\chi(\mathcal{L}(G)) \leq \omega(\mathcal{L}(G))$. Mit der trivialen Ungleichung $\chi \geq \omega$ folgt die Behauptung. □

Übung 1. *Gib einen Beweis für den Satz von König-Egerváry der mit alternierenden Pfaden arbeitet.*
Hinweis: Wähle X, Y Bipartition von G und M maximum Matching. Sei A die Menge der nicht-Matchingknoten in X und M_1 die Menge der Matchingkanten die auf alternierenden Pfaden liegen, die in Knoten aus A beginnen. Sei U die Vereinigung der Y-Knoten der Kanten von M_1 und der X-Knoten von $M - M_1$. Zeige U ist Vertex Cover von G.

Übung 2. *Entwirf einen effizienten Algorithmus der zu gegebenem M Matching und x Knoten in einem bipartiten Graphen entscheidet ob es ein Matching M' gibt das größer ist als M und in dem x ein Matchingknoten ist.*

5.4 Komplemente von Line-Graphen bipartiter Graphen

Es gelten folgende Äquivalenzen:

$$\begin{aligned} \chi(\overline{\mathcal{L}(G)}) &= \omega(\overline{\mathcal{L}(G)}) \\ \iff \kappa(\mathcal{L}(G)) &= \alpha(\mathcal{L}(G)) \\ \iff \min_{W \in VC(G)} |W| &= \max_{M \text{ Matching in } G} |M| \end{aligned}$$

Die letzte Gleichung gilt nach dem Satz von König-Egerváry. Also sind Komplemente von Line-Graphen bipartiter Graphen χ-perfekt.

Zum Kapitel

Der Satz von König–Egerváry ist eng verwandt mit anderen Min-Max Sätzen wie etwa dem Satz von Ford–Fulkerson über Flüsse in Netzwerken, dem Satz von Menger und dem Satz von Dilworth. Diese Zusammenhänge findet man etwa in Jungnickel [26]. Dort aber auch in [10] wird auch die in den Übungen angesprochene Frage nach effizienten Algorithmen zum Auffinden eines maximalen Matching behandelt.

6 Triangulierte Graphen

Zur Erinnerung: Unter einem *triangulierten* Graphen G verstehen wir einen Graphen, auf dem es keinen induzierten Kreis auf ≥ 4 Knoten gibt. (Äquivalent dazu: jeder Kreis der Länge ≥ 4 hat eine Sehne). Außerdem gilt, daß die Grapheneigenschaft trianguliert zu sein durch Obstruktionen definiert, also hereditär ist.

Beispiele. triangulierte Graphen:
- Cliquen
- Bäume
- Intervallgraphen

Definition. $v \in V$ heißt *simplizial* in G, falls $G[N(v)]$ eine Clique ist.

Satz 1. [Dirac] *Ein triangulierter Graph hat einen simplizialen Knoten.*

Um den Satz zu beweisen, benötigen wir noch ein Lemma und folgende

Definition. $S \subseteq V$ heißt *$a-b$ trennende Menge* in $G = (V, E)$, wenn es keinen Weg von a nach b in $G[V - S]$ gibt. S heißt *minimal* falls gilt: $\forall T \subset S$ enthält $G[V - T]$ einen $a, .., b$ Weg (d.h. T ist keine $a - b$ trennende Menge).

Bemerkung. Wenn a und b nicht benachbart sind, dann ist $V - \{a, b\}$ immer eine $a - b$ trennende Menge.

Lemma 2. *Sei $S \subseteq V$ eine minimale $a - b$ trennende Menge. Dann gilt:*
G trianguliert $\Rightarrow G[S]$ ist Clique.

Beweis. Angenommen es gibt $s_1, s_2 \in S$ mit $s_1, s_2 \notin E$. Seien V_a, V_b Komponenten von $G[V - S]$ mit $a \in V_a$, $b \in V_b$. Da S minimale $a - b$ trennende Menge ist existieren in $V_a \cup S$ und in $V_b \cup S$ Wege zwischen s_1 und s_2. Der von zwei solchen kürzesten Wegen induzierte Graph ist ein Kreis C. Es gilt: $C \geq 4$ und C enthält keine Sehne. Widerspruch zu G trianguliert. $\square$

9. Vorlesung, 28.11.95
Mitschrift: Carsten Ihlemann

Beweis des Satzes von Dirac. Wir zeigen folgende stärkere Behauptung: $G = (V, E)$ ist vollständig oder es existieren zwei nicht benachbarte simpliziale Knoten. Und zwar durch Induktion über $|V|$. Es gelte also die Beh. für alle Graphen kleinerer Ordnung und G sei nicht vollständig.

Seien a, b zwei nicht-benachbarte Knoten von G. Sei S eine minimale a-b-trennende Menge, nach dem Lemma ist S eine Clique. Seien V_a, V_b Komponenten von $G[V - S]$ mit $a \in V_a$, $b \in V_b$. Falls $V_a \cup S$ Clique ist, ist a selbst simplizial in G. Andernfalls hat $G[V_a \cup S]$ nach IV zwei nicht-benachbarte simpliziale Knoten. Da S Clique muß einer davon in V_a liegen, etwa c. Da $N_G(c) \supseteq N_{G[V_a \cup S]}(c)$ ist c auch simplizial in G. Analaog ex. ein simplizialer Knoten in V_b. Nach Konstruktion sind diese nicht benachbart. $\square$

Definition. Sei $G = (V, E)$ ungerichteter Graph. Eine Reihung $\Lambda = (v_1, .., v_n)$ der Knoten von G heißt *perfekte Eliminationsreihenfolge (PER)* falls v_i simplizial in $G[v_i, .., v_n]$ ist für $i = 1, .., n$.

Satz 3. *Sei $G = (V, E)$ ein ungerichteter Graph. Dann sind folgende Aussagen äquivalent.*

(1) *G ist trianguliert.*

(2) *G besitzt eine PER.*

(3) *Jede minimale trennende Menge ist Clique.*

Beweis. $(1 \Rightarrow 2)$ folgt aus dem Satz von Dirac und daraus, daß Trianguliertheit hereditär ist.
$(2 \Rightarrow 1)$ Sei Λ PER für G. Angenommen es ex. ein sehnenfreier Kreis C in G. Sei x der Knoten in C mit niedrigstem Index in Λ. Da $|N(x) \cap C| \geq 2$ muß wegen der PER Eigenschaft eine Sehne zwischen den Nachbarn von x sein $\sharp$.
$(1 \Rightarrow 3)$ Lemma 6.
$(3 \Rightarrow 1)$ Indirekt: Sei $[a, x, b, v_1, .., v_n, a]$ ein induzierter Kreis der Länge ≥ 4 in G. Sei S eine minimale a-b-trennende Menge. Dann folgt: $x \in S$ und $v_i \in S$ für ein i. Da der Kreis sehnenfrei ist, ist $v_i x$ keine Kante G, also S keine Clique. $\square$

Folgerung. Triangulierte Graphen sind in polynomialer Zeit erkennbar: Für $v \in V$ in kann in $O(|N(v)^2|)$ getestet werden ob v simplizial ist. Daher kann in $O(n^3)$ entschieden werden ob G einen simplizialen Knoten besitzt. Entferne diesen Knoten und iteriere. So bekommt man in $O(n^4)$ entweder eine PER oder einen Subgraphen der keinen simplizialen Knoten besitzt.

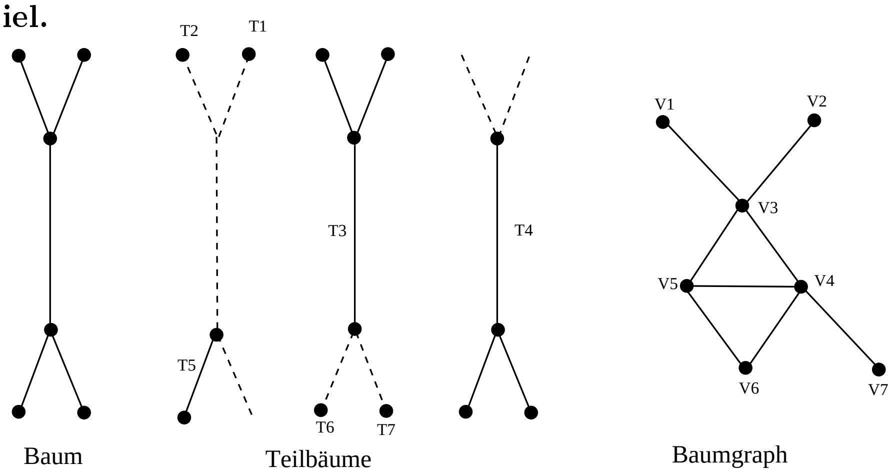
6.1 Baumgraphen

Beobachtung. Intervallgraphen lassen sich als Durchschnittsgraphen von Pfaden auf einem Pfad repräsentieren.

Übung 1. *Jeder Graph G ist Durchschnittsgraph von Subgraphen auf einem Graphen $H(G)$.*

Definition. Baumgraphen sind die Durchschnittsgraphen von Teilbäumen eines Baumes.

Beispiel.



Lemma 4. *Die Teilbäume eines Baumes T haben die Helly-Eigenschaft (d.h. sind $\{T_i\}_{i=1}^k$ paarweise nicht disjunkte Teilbäume von T, dann gilt $\bigcap_{i=1}^k T_i \neq \emptyset$.)*

Beweis. Seien $T_1, .., T_k$ Teilbäume von T mit $T_i \cap T_j \neq \emptyset$ für alle i, j. Sei $v \in V(T)$ so gewählt, daß die Indexmenge $\{j \in \{1..k\} : v \in V(T_j)\}$ maximal ist. Angenommen es ex. ein Baum T_i, der v nicht enthält. Sei $w \in T_i$ so gewählt, daß der Abstand in T zwischen v und w minimal ist. Für alle T_j mit $v \in T_j$ ex. ein $v_j \in T_j \cap T_i$. Da T_j zusammenhängend ist, enthält T_j den (eindeutigen) Weg von v_j nach v (in T), und dieser Weg enthält w. Also ist w in mehr Bäumen als v enthalten $\sharp$. $\triangle$

Satz 5. *Äquivalent sind:*

(1) *G ist trianguliert*

(2) *G ist Baumgraph*

(3) *Es gibt einen Baum $T_{\mathcal{C}} = (\mathcal{C}, \Sigma)$, wobei $\mathcal{C}$ die Menge der maximalen Cliquen von G ist, so daß für alle $v \in V$ gilt: $T_v := T[\{C \in \mathcal{C} : v \in C\}]$ ist zusammenhängend und also Subbaum.*

Beweis. ($1 \Rightarrow 3$) Induktion über $|V|$. Falls G vollständig ist, so besteht $T_{\mathcal{C}}$ nur aus einem Punkt. Sei also im Folgendem $G \neq K_n$, Es gibt also nichtbenachbarte Knoten a, b in G. Da G trianguliert ist gibt es eine $a-b$-trennende Clique C. Sei W eine Zusammenhangskomponente von $G[V-C]$ sowie $G^1 = G[V-W]$, $G^2 = G[W+C]$. Seien $T_{\mathcal{C}^1}$ und $T_{\mathcal{C}^2}$ die nach Induktionsvoraussetzung existierenden Bäume für die Graphen G^1 und G^2. In $\mathcal{C}^1$ und $\mathcal{C}^2$ gibt es Cliquen C^1 und C^2 die C umfassen. Gilt $C \neq C^1, C^2$ dann erhalten wir $T_{\mathcal{C}}$ indem wir $T_{\mathcal{C}^1}$ und $T_{\mathcal{C}^2}$ durch eine Kante zwischen C^1 und C^2 verbinden. Gilt $C = C^1$ oder $C = C^2$ dann erhalten wir $T_{\mathcal{C}}$ indem wir die Knoten C^1 und C^2 aus $T_{\mathcal{C}^1}$ und $T_{\mathcal{C}^2}$ verschmelzen. Der so entstandene Knoten entspricht der maximalen Clique $C' = C^1 \cup C^2$ aus G (natürlich ist $C' = C^1$ oder $C' = C^2$).

($3 \Rightarrow 2$) Der Zugrundeliegende Baum sei $T = T_{\mathcal{C}}$. Zu $v \in V$ gehört der Subbaum T_v von V. Das ist eine Repräsentation von G weil: $v, w \in E \iff$ ex. max. Clique C mit $v, w \in C \iff T_v \cap T_w \neq \emptyset$.

($2 \Rightarrow 1$) Sei $\{T_v\}_{v \in V}$ eine Familie von Subbäumen eines Baumes $T = (A, F)$, derart daß uv Kante in G gdw: $T_v \cap T_w \neq \emptyset$.

Behauptung: G hat simplizialen Knoten.
Sei $a_1, a_2, ..., a_t$ eine Reihung der Knoten von T so daß a_1 ein Blatt von T ist und alle Subgraphen $T[a_1, .., a_i]$ zusammenhängend sind. Sei i_0 das kleinste i für das $T[a_1, .., a_i]$ einen der Bäume T_v enthält und sei $T_{v_0} \subseteq T[a_1, .., a_{i_0}]$ enthalten. Für jedes w mit $vw \in E$ gilt $T_w \cap T[a_1, .., a_{i_0}] \neq \emptyset$ und $T_w \not\subseteq T[a_1, .., a_{i_0}]$, also $a_{i_0} \in T_w$. Daher ist $N[v_0] = \{w : a_{i_0} \in T_w\}$ eine Clique in G. □

Übung 2. *G trianguliert, C max. Clique und $\{T_i\}_{i \in V}$ Baumdarstellung von G auf dem Baum T. Dann gibt es einen Knoten $a \in T$, der C entspricht (d.h. $\{i \in V : a \in T_i\} = C$).*

Übung 3. *G sei ein Graph, in dem die Pfade die Helly-Eigenschaft haben, dann gilt: G ist ein Baum.*

Übung 4. *G trianguliert $\iff \mathcal{L}(G)$ trianguliert.*

6.2 Erkennung von triangulierten Graphen

10. Vorlesung, 30.11.95
Mitschrift: Stefanie Gerke

Sei $G = (V, E)$ ein Graph und $\Theta : V \longrightarrow \{1 \dots n\}$ (bijektiv) eine Reihung der Knoten von G, das Paar $(v, w) \in V \times V$ nennen wir Θ–*Haken*, wenn folgende Bedingungen erfüllt sind:

(1) $\Theta(v) < \Theta(w)$ und $\{v, w\} \notin E$.

(2) Es existiert ein *Zeuge* $u \in V$ für (v, w), das ist ein Knoten von dem aus Θ aufsteigende Pfade $u = v_0, v_1, .., v_s = v$ und $u = w_0, w_1, .., w_t = w$ in G existieren, d.h., aus $i < j$ folgt $\Theta(v_i) < \Theta(v_j)$ und $\Theta(w_i) < \Theta(w_j)$. Zusätzlich verlangen wir noch $\Theta(w_{t-1}) < \Theta(v)$.

Die Menge $F(\Theta) = \Big\{\{v, w\} : v, w \in V; \{v, w\} \notin E;\ (v, w) \text{ ist ein } \Theta\text{–Haken}\Big\}$ heißt *fill-in* von G bezüglich der Reihung Θ. Sei $E_\Theta = E \cup F(\Theta)$ und $G_\Theta = (V, E_\Theta)$.

Lemma 6. *$G = G_\Theta \Longleftrightarrow \Theta$ ist PER für G.*

Beweis. ($\Rightarrow$) $G = G_\Theta \Rightarrow$ Für alle $u \in V$ mit $\Theta(u) < \Theta(w), \Theta(v)$ und $\{u, v\}, \{u, w\} \in E$ gilt, daß $\{v, w\} \in E$ ist (andernfalls wäre u ein Zeuge für den Haken (v, w)) $\Rightarrow$ die rechte Nachbarschaft von u bzgl. Θ ist eine Clique $\Rightarrow \Theta$ ist PER.

($\Leftarrow$) Angenommen $G \neq G_\Theta \Rightarrow$ Es existiert ein lexikographisch kleinster Θ–Haken (v, w), d.h. daß für alle anderen Θ–Haken (v', w') gilt: $\Theta(v) < \Theta(v')$ oder $\Theta(v) = \Theta(v')$ und $\Theta(w) < \Theta(w')$. Für diesen Haken wählen wir nun den Θ maximalen Zeugen u und Pfade $u = v_0, v_1, .., v_s = v$ und $u = w_0, w_1, .., w_t = w$ wie in der Definition.

Behauptung $v_1 = v$. *Bew:* Sei i minimal so daß $\Theta(w_i) > \Theta(v_1)$. Aus $v_1 \neq v$ und $\{v_1, w_i\} \notin E$ folgt, daß (v_1, w_i) ein lexikographisch kleinerer Haken ist als (v, w). Aus Aus $v_1 \neq v$ und $\{v_1, w_i\} \in E$ folgt, daß v_1 ein größerer Zeuge für (v, w) ist als u.

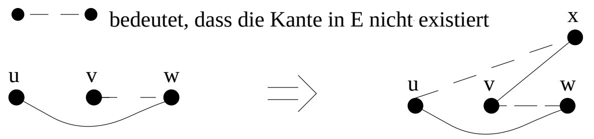
Behauptung $w_{t-1} = u$. *Bew:* Aus $w_{t-1} \neq u$ und $\{w_{t-1}, v\} \notin E$ folgt daß (w_{t-1}, v) ein lexikographisch kleinerer Haken ist als (v, w). Aus $w_{t-1} \neq u$ und $\{w_{t-1}, v\} \in E$ folgt daß w_{t-1} ein größerer Zeuge für (v, w) ist als u.

Also sind $v, w \in N(u)$ und beide größer als u bzgl. Θ, aber sie sind nicht mit Hilfe einer Kante in G verbunden. Dies ist aber ein Widerspruch zur PER

Eigenschaft von Θ bei u. □

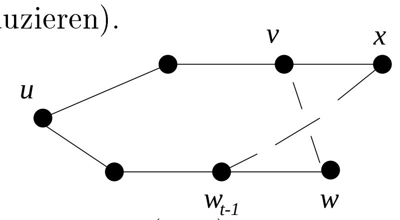
Folgerung. G_Θ ist immer ein triangulierter Graph.

Lemma 7. *Sei G ein triangulierter Graph und Θ eine Reihung mit der Eigenschaft ($\star$), daß aus $\Theta(u) < \Theta(v) < \Theta(w)$; $\{v,w\} \notin E$ und $\{u,w\} \in E$, die Existenz eines Knotens x folgt, der bezüglich Θ größer ist als v und in dessen Nachbarschaft v, aber nicht u liegt (siehe Bild), dann ist Θ eine PER.*



Beweis. Angenommen Θ wäre keine PER $\Rightarrow G \neq G_\Theta \Rightarrow$ Es existiert ein Θ-Haken. Sei (v,w) ein Θ-Haken mit $\Theta(v)$ maximal.

Wegen $\Theta(w_{t-1}) < \Theta(v)$ folgt aus der Eigenschaft ($\star$) die Existenz eines Knotens x, so daß $\Theta(v) < \Theta(x)$, $\{v,x\} \in E$, aber $\{w_{t-1},x\} \notin E$ gilt. Da G als trianguliert vorausgesetzt wurde, kann $\{x,w\}$ keine Kante von G sein (andernfalls würden einige der Knoten v_i und w_i zusammen mit x einen sehnenlosen Kreis der Länge ≥ 4 induzieren).



Wenn $\Theta(x) < \Theta(w)$ dann ist (x,w) ein Θ-Haken mit $\Theta(x) > \Theta(v)$ im Widerspruch zur Wahl von (v,w). Wenn $\Theta(w) < \Theta(x)$ dann ist (w,x) ein Θ-Haken mit $\Theta(w) > \Theta(v)$ im Widerspruch zur Wahl von (v,w). □

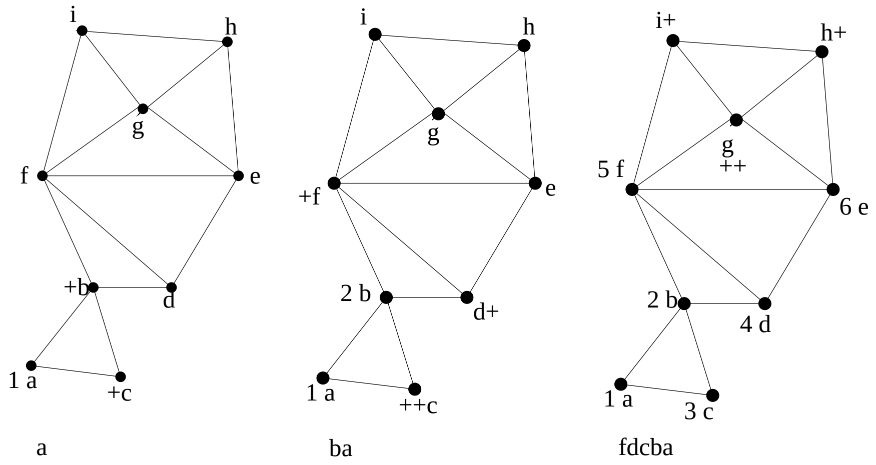
6.2.1 Maximum Cardinality Search

Wir wollen entscheiden, ob ein gegebener Graph G trianguliert ist. Dazu bestimmen wir zuerst eine Reihung Θ der Knoten aus V, so daß diese eine PER ist, wenn G ein triangulierter Graph ist. In einem zweiten Schritt wird dann geprüft, ob $G = G_\Theta$ ist, was (Lemma 10 und Satz 7) impliziert, daß G trianguliert ist. Die Reihung berechnen wir mit MaxCardSearch.

MaxCardSearch(G)

$\Theta \leftarrow [\,]$
for all $v \in V$ **do**
  $\alpha(v) \leftarrow 0$
while $V \neq \emptyset$
  choose v with $\alpha(v)$ maximal
  for all $w \in N(v)$ **do**
    $\alpha(w)$++
  remove v from V
  $\Theta \leftarrow v + \Theta$
return Θ

Beispiel. Das folgende Abbildung zeigt drei Bilder eines Graphen G mit Knoten $a, b, .., i$ während der Ausführung von MaxCardSearch auf G. Das erste Bild zeigt den Graphen nachdem a als erster Knoten gewählt wurde, a hat die Nummer 1 bekommen und α von b und c haben jetzt den Wert 1 den wir durch das + andeuten. Als zweiter Knoten darf nun entweder b oder c gewählt werden. Das zweite Bild zeigt den Zustand nach der Wahl von b. Als drittes muß c gewählt werden. Das dritte Bild zeigt den Graphen nachdem die ersten sechs Knoten in der durch $\Theta = efdcba$ beschriebenen Reihenfolge gewählt sind.



Diese Prozedur hat die Laufzeit $O(n + \sum d_i)$, wenn das Element mit maximalem Wert α in O(1) gefunden wird, bzw. bei allen Schleifendurchläufen nur $O(n + m)$ Zeit benötigt wird.

Einschub: Eine Datenstruktur für α

Sei A ein array $[1..n]$ mit $A[i] = \{v : \alpha(v) = i\}$ und t der Maximale Index i für den $A[i]$ nicht leer ist. Jedes $A[i]$ ist als doppelt verkettete Liste realisiert. Zu Beginn der Prozedur ist $A[1] = V$ und $t = 1$. Durch einen Zeiger beim Knotenelement $v$können wir direkt auf das Element v in der Liste $A[\alpha(v)]$ zugreifen. So ist das Löschen von v aus $A[\alpha(v)]$ in konstanter Zeit möglich, auch das Inkrementieren von $\alpha(v)$ bereitet keine Probleme. Die Verwaltung von t bereitet nur ein Problem. Wenn $A[t]$ nach dem Löschen eines Elements leer ist, so muß t vermindert werden bis $A[t]$ mindestens ein Element enthält, dies ist im allgemeinen nicht in konstanter Zeit möglich. Die Gesamtzahl der Verminderungen ist aber durch die Gesamtzahl der Erhöhungen beschränkt. Der Gesamtaufwand kann also abgeschätzt werden durch doppelte Zählung der Erhöhungen. Diese *amortisierte Aufwandsabschätzung* liefert als Ergebnis eine Laufzeit von $O(n + m)$.

Lemma 8. *Wenn G ein triangulierter Graph ist und Θ das Ergebnis von* MaxCardSearch(G), *dann ist Θ eine PER.*

Beweis. z.z Θ hat die Eigenschaft $(\star)$. Es sei $\Theta(u) < \Theta(v) < \Theta(w)$ und $\{u, w\} \in E$, $\{v, w\} \notin E$, betrachte den Zeitpunkt zu dem v genommen wurde (d.h. $\alpha(v)$ war maximal): Bei $\alpha(u)$ wurde w gezählt, bei $\alpha(v)$ nicht. Da aber $\alpha(v)$ maximal ist, muß es mindestens einen Knoten x geben, der in der Nachbarschaft von v, aber nicht in der Nachbarschaft von u liegt, und bezüglich Θ größer ist als v. $\square$

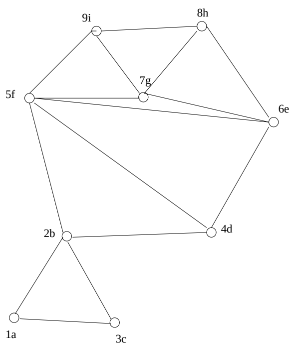
6.2.2 Erkennung perfekter eliminations Reihungen

11. Vorlesung, 05.12.95
Mitschrift: Bernd Falkenberg

Der Algorithmus MaxCardSearch produziert eine PER (perfekte Eliminationsreihe), wenn G trianguliert ist. Wie kann man jedoch erkennen, ob eine gegebene Reihung Θ der Knoten von G eine PER ist? Dazu betrachten wir den folgenden Algorithmus :

```
PER(Θ,G)
  for i = 1 to n
      w ← i-ter Knoten aus Θ
      index[w]=i
      s[w]← w
      for v ∈ N(w) with v <_Θ w
          if s[v]=v then s[v]← w
          index[v]← i
      for v ∈ N(w) with v <_Θ w
          if index[s[v]] < i then reject
  accept
```

Idee des Algorithmus: Der Wert s(v) gibt für jeden Knoten v an, welcher Knoten der nächstgrößere (bezüglich der Reihung Θ) Nachbarknoten von v ist. Der Wert index(v) gibt den Knoten an, zu dem v benachbart ist und der unter den schon bearbeiteten Knoten möglichst Θ-groß ist, d.h. möglichst weit rechts in Θ steht. Wenn der Fall eintritt, daß der Knoten s(v) eines Nachbarn v des gerade bearbeiteten Knotens w einen niedrigeren Indexeintrag als w hat, heißt das, daß der Knoten s(v) nicht zu w benachbart ist. Somit hätte man nachgewiesen, daß die rechte Nachbarschaft von v in Θ keine Clique ist. Das bedeutet, daß Θ kein PER ist und der Algorithmus bricht mit 'reject' ab.



Sei $\Theta = ihgfedcba$, die nachfolgende Tabelle beschreibt die Zustände von PER(Θ,G) für den im Bild dargestellten Graphen G. In der ersten Spalte ist der derzeit bearbeitete Knoten zu finden, in der zweiten die jeweils aktuellen Indexwerte der Knoten und in der dritten die s-Zuweisungen

:	$i\ h\ g\ f\ e\ d\ c\ b\ a$	
i :	1	$s(i) \leftarrow i$
h :	2 2	$s(i) \leftarrow h$, $s(h) \leftarrow h$
g :	3 3 3	$s(h) \leftarrow g$, $s(g) \leftarrow g$
f :	3 4 4 4	$s(g) \leftarrow f$, $s(f) \leftarrow f$
e :	5 4 5 5 5	$s(f) \leftarrow e$, $s(e) \leftarrow e$

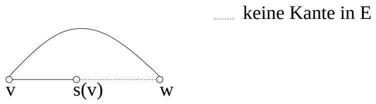
In der letzten Zeile ist ein Abbruch des Algorithmus festzustellen. Der aktuelle

Knoten ist e, $i \in N(e)$ und $s[i] = h$ aber index$[h] < 5 =$index$[e]$. Also ist die rechte Nachbarschaft von i keine Clique und Θ keine PER.

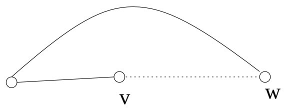
Satz 9. $\textsc{Per}(\Theta, G)$ *liefert als Ergebnis 'accept'* $\Leftrightarrow$ Θ *ist eine PER für* G.

Beweis. ($\Leftarrow$) durch Kontraposition.

Sei $\textsc{Per}(\Theta, G) =$ reject. Dann gibt es $\{v, w\} \in E$ so daß keine Kante zwischen s(v) und w ist. Also ist Θ keine PER, da v nicht simplizial in der Menge der Knoten mit Θ-Werten $\geq \Theta(v)$ ist.



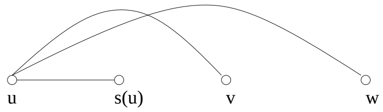
($\Rightarrow$) Sei $\textsc{Per}(\Theta, G) =$ accept. Angenommen, Θ ist keine PER, d.h. es gibt einen Knoten u, so daß u nicht simplizial in der Menge der Knoten mit Θ-Werten $\geq \Theta(u) \Rightarrow$ es gibt mindestens eine 'fehlende' Kante.



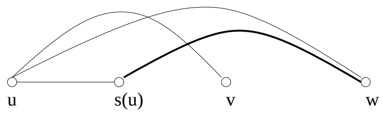
Wir sortieren die fehlenden Kanten von G lexikographisch. Sei $e = (v, w)$ die lexikographisch kleinste fehlende Kante. Wähle u maximal $\leq_\Theta$ als v und w so daß (u, v) und (u, w) Kanten sind.

Behauptung: $v = s(u)$. Indirekt angenommen $v \neq s(u)$:

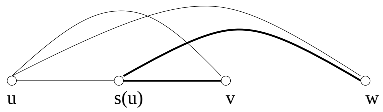
Aus der Definition von $s(u)$ folgt: $s(u) \leq_\Theta v$



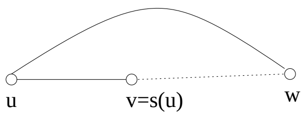
Da e die lexikalisch kleinste fehlende Kante ist, gilt $(s(u), w) \in E$



und ebenso $(s(u), v) \in E$



$\Rightarrow$ Widerspruch zur Maximalität von u. Also $v = s(u)$.



Nun gilt: Beim Besuch von w ist index(s(u)) $<$ index(w), also bricht PER(Θ, G) mit reject ab. Widerspruch. □

6.3 Triangulierte Graphen und ihre Komplemente sind χ-perfekt.

Zunächst erinnern wir uns an den Färbungsalgorithmus GREEDY (vergl. Abschnitt 2.3, Färbung von Intervall Graphen), der für einen gegebenen Graphen und einer Reihung seiner Knoten eine Färbung der Knoten liefert.

GREEDY(Θ)
  $R \leftarrow \emptyset$
  while $\Theta \neq \emptyset$
    $v \leftarrow$first(Θ)
    $c \leftarrow 1$
    while $\exists w \in (N(v) \cap R)$ with col(w) $= c$
      c++
    col(v) $\leftarrow c$
    $R \leftarrow R + v$
    $\Theta \leftarrow \Theta - v$
  return col

Lemma 10. *Sei G trianguliert, $\Theta = v_1, ..., v_n$ eine PER für G, dann liefert* GREEDY($\overleftarrow{\Theta}$) $=$ GREEDY($v_n, v_{n-1}, ..., v_1$) *eine optimale Färbung.*

Beweis. Wird v_i mit Farbe j gefärbt, dann ist v_i in einer j-Clique enthalten. Das gilt, weil v_i simplizial in der Menge der zur Färbungszeit von v_i schon gefärbten Knoten ist (denn: Θ ist eine PER für G). $\Rightarrow$ Wenn Greedy k Farben benutzt, dann enthält G eine k-Clique, d.h. die Färbung ist optimal. □

Aus diesem Lemma ergibt sich sofort der folgende Satz:

Satz 11. *Triangulierte Graphen sind χ-perfekt.*

Beweis. Trianguliertheit ist eine hereditäre Eigenschaft(d.h. wenn ein Graph trianguliert ist, sind auch sämtliche induzierte Untergraphen trianguliert), daher genügt es zu zeigen, daß für einen triangulierten Graphen gilt: $\chi(G) = \omega(G)$. Da triangulierte Graphen eine PER besitzen, folgt mit Lemma 12 die Behauptung. $\square$

Übung 5. *Wie kann* Greedy *implementiert werden, so daß die Laufzeit nur $O(n + m)$ ist?*

Wir wollen zeigen, daß die Komplemente von triangulierten Graphen auch χ-perfekt sind. Dazu benutzen wir wieder die bekannte Äquivalenz

G ist α-perfekt $\Leftrightarrow$ $\overline{G}$ ist χ-perfekt

Wir betrachten nun eine Art Greedy-Algorithmus zur Bestimmung von unabhängigen Mengen:

GreedyIndSet(Θ)
  $I \leftarrow \emptyset$
  repeat
    $v \leftarrow$first(Θ)
    $C_v \leftarrow (N(v) \cap \Theta)$
    remove $C_v + v$ from Θ
    $I \leftarrow I + v$
  until $\Theta = \emptyset$
  return I

Lemma 12. *Sei G trianguliert, Θ PER für G uns sei I die von* GreedyIndSet(Θ) *zurückgegebene Menge. I ist eine maximum unabhängige Menge von G.*

Beweis.

(1) I ist unabhängig $\surd$

(2) Wir betrachten die Menge $K := \{v + C_v : v \in I\}$

- Für jedes v ist $v + C_v$ eine Clique (PER-Bedingung)
- Weil Θ zum Schluß leer ist, ist K eine Cliquenzerlegung

- Aus $|K| = |I|$ folgt wegen $\kappa(G) \geq \alpha(G)$, daß I maximum unabhängige Menge und K minimum Cliquenzerlegung ist. □

Satz 13. *Komplemente von triangulierten Graphen sind χ-perfekt.*

Beweis. Es gilt: $\overline{G}$ ist χ-perfekt $\Leftrightarrow$ G ist α-perfekt. Trianguliertheit ist eine hereditäre Eigenschaft daher ist auch die Klasse der Komplemente von triangulierten Graphen eine hereditäre Graphenklasse. Aus dem Lemma wissen wir $\alpha(G) = \kappa(G)$ wenn G das Komplement eines triangulierten Graphen ist. □

6.4 Perfekte Reihungen

Definition. Eine Reihung $\Theta = v_1, ..., v_n$ von V heißt *perfekte Reihung*, wenn für alle $W \subseteq V$ gilt: Greedy($\Theta|_W$) liefert eine optimale Färbung von $G[W]$.

Ein Graph G heißt *perfectly orderable* $\iff$ es existiert eine perfekte Reihung der Knoten von G.

Bemerkung.

- Triangulierte Graphen besitzen eine perfekte Reihung der Knoten (jede PER hat die geforderte Eigenschaft).
- Vergleichbarkeitsgraphen besitzen eine perfekte Reihung der Knoten (jede lineare Erweiterung (topologische Sortierung) hat die geforderte Eigenschaft).

Satz 14. *Θ ist perfekte Reihung $\iff$ es gibt keinen schlechten P_4, d.h. es existieren keine vier Knoten u, v, x, y mit folgenden Eigenschaften:*

- $(u, v) \in E,\ (v, x) \in E,\ (x, y) \in E$
- $(x, u) \notin E,\ (u, y) \notin E,\ (y, v) \notin E$
- *$u <_\Theta v$ und $x >_\Theta y$*

Beweis. ($\Rightarrow$) durch Kontraposition. Annahme: Es gibt einen schlechten P_4 in G. Betrachte $G|_{u,v,x,y}$, wobei u, v, x, y die Knoten des schlechten P_4 seien. Der Greedy-Algorithmus wird u und y die gleiche Farbe geben, da zwischen u und y keine Beziehung besteht. Damit folgt allerdings, daß für die Knoten v

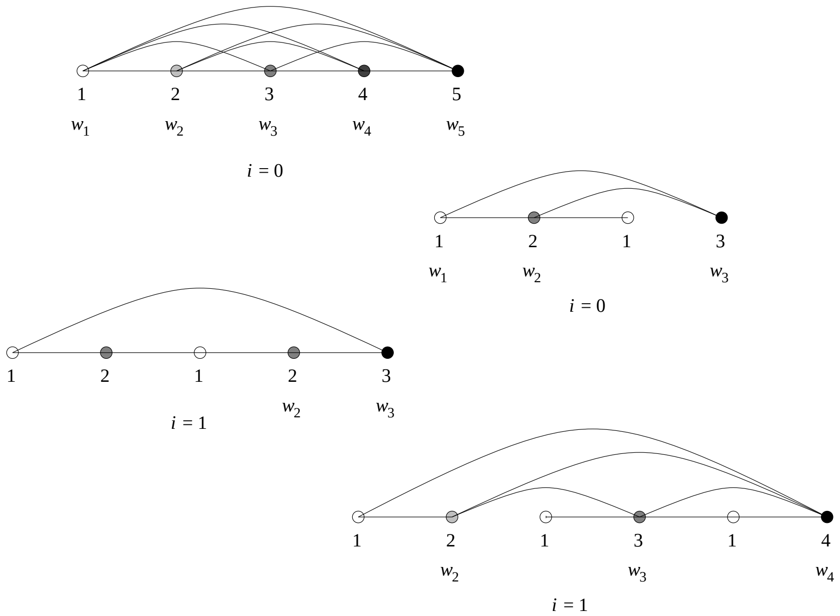
und x noch zwei Farben benötigt werden, da v und x benachbart sind. Somit ergibt sich, daß der Greedy-Algorithmus drei Farben benötigt, um diesen P_4 zu färben. Notwendig sind aber nur 2 Farben, da man u und x sowie v und y die gleiche Farbe geben kann. Damit folgt, daß Θ keine perfekte Reihung sein kann.

12. Vorlesung, 12.12.95
Mitschrift: Johannes Schwarz

$(\Longleftarrow)$
Behauptung: Sei Θ eine Reihung der Knoten des Graphen G, die keinen schlechten P_4 enthält. Dann liefert Greedy(Θ) eine optimale Färbung von G.

Beweis durch Kontraposition: Sei G durch Greedy(Θ) k-gefärbt, mit $k > \chi(G)$; sei i die kleinste Farbe, so daß es eine Clique $w_{i+1}, \ldots, w_k$ gibt mit $\mathrm{col}[w_j] = j$. (Wenn $E(G) \neq \emptyset$, ist $k \geq 2$, und dann ist sicher $i \leq k-2$.)

Bsp.:

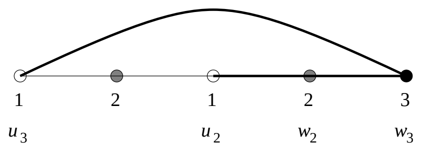


Beobachtung: Wäre $i = 0$, dann enthielte G eine k-Clique, so daß gälte: $k \leq \omega(G) \leq \chi(G)$ (#).

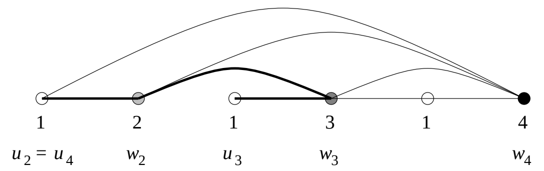
Also muß $i > 0$ sein, und dann enthält Θ einen schlechten P_4, was man so einsieht:

Für alle $j \in \{i+1, \ldots, k\}$ wähle $u_j <_\Theta w_j$ so, daß $\mathrm{col}[u_j] = i$ und $u_j w_j \in E$

ist. Ein solches u_j muß existieren, denn anderenfalls hätte w_j eine Farbe $\leq i$ bekommen. Falls es mehrere solcher u_j gibt, so wähle das Θ-kleinste.

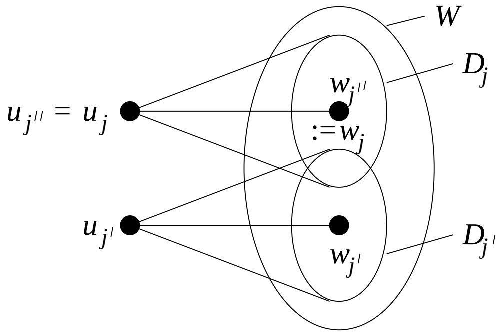


$i = 1$

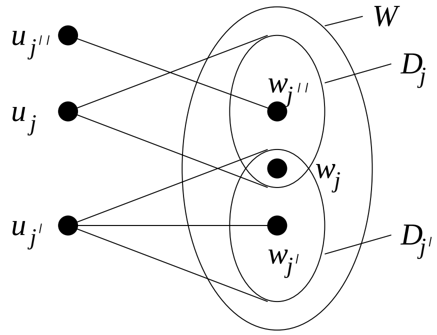


$i = 1$

Definiere dann $U := \{u_j : j = i+1, ..., k\}$ und $W := \{w_j : j = i+1, ..., k\}$ sowie $D_j := N(u_j) \cap W$. Sei j so, daß $|D_j| \geq |D_{j'}|$ f. a. $j' \in i+1, ..., k$. Wäre $|D_j| = |W|$, dann wäre $\{u_j\} \cup W$ Clique, d. h., wir hätten eine Clique $w_i := u_j, w_{i+1}, ..., w_k$ bei der Wahl von i gehabt (# zur Minimalität von i). Also ist $|D_j| < |W|$; sei $w_{j'} \in W \backslash D_j$. Aus $|D_{j'}| \leq |D_j|$ und $w_{j'} \in D_{j'} \backslash D_j$ folgt $D_j \backslash D_{j'} \neq \emptyset$; sei also $w_{j''} \in D_j \backslash D_{j'}$, wobei $w_{j''} := w_j$, falls $w_j \notin D_{j'}$.



1. Fall: $w_j \notin D_{j'}$



2. Fall: $w_j \in D_{j'}$

Könnten wir jetzt zeigen, daß der P_4 $u_j - w_{j''} - w_{j'} - u_{j'}$ schlecht ist, so wären wir fertig; das Problem ist jedoch, daß u_j nicht immer Θ-kleiner als $w_{j''}$ ist. Also nehmen wir statt u_j den Knoten $u_{j''}$ und haben nun noch zu zeigen, daß $u_{j''} - w_{j''} - w_{j'} - u_{j'}$ ein schlechter P_4 ist. Dazu ist zu zeigen:

(a) Die drei Kanten $u_{j''}w_{j''}$, $w_{j''}w_{j'}$ und $w_{j'}u_{j'}$ existieren in G.

(b) Die drei Kanten $u_{j''}w_{j'}$, $u_{j'}w_{j''}$ und $u_{j''}u_{j'}$ existieren *nicht* in G.

(c) $u_{j''} <_\Theta w_{j''}$ und $u_{j'} <_\Theta w_{j'}$. $\surd$

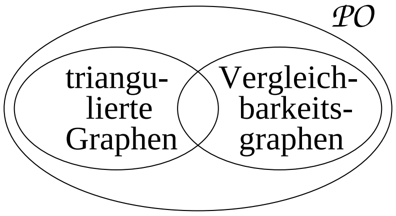
Zu (a): $w_{j''}w_{j'} \in G$, da W Clique ist.

Zu (b): Zum ersten ist im Falle $w_j \notin D_{j'}$ der Knoten $u_{j''}$ ja u_j, und dann folgt aus $w_{j'} \notin D_j$, daß $w_{j'} \notin N(u_j)$; im Falle $w_j \in D_{j'}$ gilt $u_j <_\Theta u_{j'}$, weil sonst u_j

auf den Knoten $u_{j'} \neq u_j$ definiert worden wäre, und $u_{j''} <_\Theta u_j$, weil sonst $u_{j''}$ auf den Knoten $u_j \neq u_{j''}$ definiert worden wäre, folglich gilt auch $u_{j''} <_\Theta u_{j'}$; wäre nun $u_{j''}w_{j'} \in G$, so wäre mithin $u_{j'}$ auf den Knoten $u_{j''}$ definiert worden; also ist in beiden Fällen $u_{j''}w_{j'} \notin G$. Zum zweiten folgt aus $w_{j''} \notin D_{j'}$, daß $w_{j''} \notin N(u_{j'})$, also ist $u_{j'}w_{j''} \notin G$. Zum dritten haben $u_{j''}$ und $u_{j'}$ beide Farbe i, also ist $u_{j''}u_{j'} \notin G$.

Enthält nun Θ keinen schlechten P_4, so auch nicht eine Einschränkung von Θ, und dann folgt aus der Behauptung, daß $k = \chi(G')$ für alle Subgraphen G' von G ist, und somit ist Θ perfekt. □

Die Klasse der Graphen, die eine perfekte Reihung besitzen, heißt *perfectly orderable* ($\mathcal{PO}$).

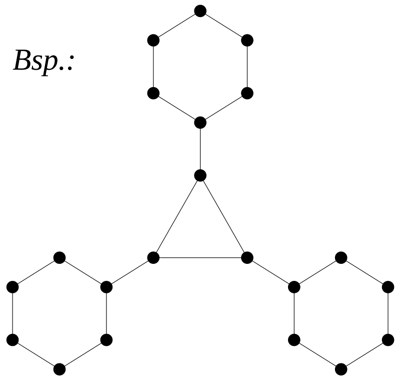


Satz 15. *$\mathcal{PO}$–Graphen sind χ-perfekt.*

Beweis. Sei G ein $\mathcal{PO}$–Graph und Θ eine perfekte Reihung von G, so enthält Θ keinen schlechten P_4. Es seien k und i wie im Beweis der Rückrichtung des vorhergehenden Satzes definiert. Wäre $i > 0$, dann enthielte, wie oben gezeigt, Θ einen schlechten P_4 (#). Also ist $i = 0$, d.h., G enthält eine k-Clique, so daß $\omega(G) \geq k \geq \chi(G)$ und wegen $\omega(G) \leq \chi(G)$ mithin $\omega(G) = \chi(G)$ gilt. Da mit Θ auch jede Einschränkung von Θ eine perfekte Reihung ist, folgt die χ-Perfektheit von G. □

Übung 6. *Zeige direkt: Lineare Erweiterungen von Ordnungen und PER von triangulierten Graphen enthalten keinen schlechten P_4.*

Übung 7. *Es gibt χ-perfekte Graphen, die keine $\mathcal{PO}$-Graphen sind.*



Bemerkung Erkennung von $\mathcal{PO}$-Graphen ist ein NP–vollständiges Problem.

Zum Kapitel

Nimmt man zu den in den letzten beiden Kapiteln vorgestellten Klassen noch Vergleichbarkeitsgraphen und Unvergleichbarkeitsgraphen sowie einige Teilklassen wie Intervallgraphen und Permutationsgraphen dazu, so hat man die typische Klassen perfekter Graphen beisammen (vergl etwa Lovász [30]). Wir verschieben eine gründliche Behandlung von Vergleichbarkeitsgraphen bis Kapitel 9.

Triangulierte Graphen standen ganz am Anfang der Theorie perfekter Graphen. Hajnal und Surányi zeigten 1958 die α-Perfektheit und Berge 1960 die χ-Perfektheit dieser Klasse. Diese Ergebnisse standen Pate bei der Formulierung der *weak perfekt graph conjecture* (dem jetzigen Perfekt Graph Theorem) durch Berge 1961. Der hier vorgestellte Erkennungsalgorithmus für triangulierte Graphen ist von Tarjan und Yannakakis [47]. Es gibt noch einen zweiten linearzeit Algorithmus für dieses Problem (vergl etwa [18].

Triangulierte Graphen wurden in verschiedenste Richtungen verallgemeinert. Gallai wies die Perfektheit von Graphen in denen jeder ungerade Kreis zwei nichtkreuzende Sehnen hat nach. Olaru und Sachs zeigten die Perfektheit der Graphen in denen jeder ungerade Kreis zwei kreuzende Sehnen hat. Schliesslich zeigte Meyniel: Jeder Graph in dem jeder ungerade Kreis zwei Sehnen hat ist Perfekt. Diese Graphen werden *Meyniel–Graphen* genannt (Die Originalarbeit ist in [2] wiedergegeben, einen alternativen Beweis gibt [30]). Burlet und Fontlup definierten eine Kompositionsoperation (Amalgamierung) für Meyniel–Graphen und konnten basierend darauf zeigen, daß Meyniel–Graphen in polynomieller Zeit erkannt werden können. Die Originalarbeit ist in [2] nachgedruckt.

Hayward [23] definierte *weakly chordal graphs* als Graphen G mit der Eigenschaft: Es gibt keine induzierten Kreise der Länge ≥ 5 in G und in $\overline{G}$. Weakly chordal graphs sind perfekt und können in polynomieller Zeit erkannt werden [24].

Eine weitere Linie von Verallgemeinerungen triangulierter Graphen führt auf Hypergraphen mit zugrundeliegender Baumstruktur. Quellen hierzu sind [47] und [5].

Zuletzt sei noch der Begriff der *partiellen k-Bäume* erwähnt. Das sind Graphen G die sich in einen triangulierten Graphen $H(G)$ mit $\omega(H(G)) \leq k$ einbetten lassen. Solche Einbettungen können für festes k in linearer Zeit gefunden werden (diese Algorithmen existieren jedoch nur in der Theorie). Ist eine Einbettung von G in $H(G)$ gegeben, dann lassen sich viele Probleme die für allgemeine Graphen NP-schwer sind für G effizient lösen. Quellen hierzu sind [4] und [27].

$\mathcal{PO}$–Graphen wurden von Chvátal eingeführt [2]. Die NP-Vollständigkeit der Erkennung ist von Middendorf und Pfeiffer [36].

Einen Überblick über eine Vielzahl weiterer Graphenklassen und ein umfangreichen Literaturverzeichnis bietet Brandstädt [5]. Dieser Arbeit liegt auch eine Skizze der Inklusionsrelation für 120 Klassen perfekter Graphen bei.

7 Polytope und unabhängige Mengen

7.1 Ungleichungen für unabhängige Mengen

Sei $\{e_v\}_{v\in V}$ Basis des $\mathbb{R}^{|V|}$. Dann heißt $\mathbb{1}_A := \sum_{v\in A} e_v$ der *charakteristische Vektor* von A.

Sei $M = \{x_v\}_{v\in V}$ eine Menge von Vektoren, die konvexe Hülle von M ist die Menge

$$\operatorname{conv}(M) := \{\textstyle\sum_{v\in V} \lambda_v x_v : \sum_{v\in V} \lambda_v = 1 \text{ und } \lambda_v \geq 0 \quad \forall v \in V\}.$$

Sei nun $G = (V, E)$ ein Graph mit einer Abbildung $w : V \longrightarrow \mathbb{Q}_+$. Für Mengen $A \subseteq V$ definieren wir $w(A) := \sum_{v\in A} w(v) = w^\top \cdot \mathbb{1}_A$. Die *gewichtete Unabhängigkeitszahl* von G bezüglich w ist

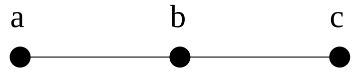
$$\alpha(G, w) := \max\{w(I) : I \subseteq V \text{ unabhängig}\}.$$

Da $\alpha(G, \mathbb{1}) = \alpha(G)$ ist die Bestimmung von $\alpha(G, w)$ i.a. NP-hart.

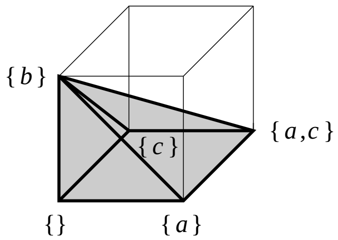
Definition. Das *Polytop der unabhängigen Mengen* eines Graphen G sei die Menge

$$\text{STAB}(G) := \text{conv}(\{\mathbb{1}_I \in \mathbb{R}^{|V|} : I \subseteq V \text{ unabh.}\}).$$

Beispiel.



Menge der unabhängigen Mengen: $\Big\{\emptyset, \{a\}, \{b\}, \{c\}, \{a, c\}\Big\}$



Lemma 1. $\alpha(G, w) = max\{w^\top \cdot x : x \in \text{STAB}(G)\}$

Beweis. ($\leq$) Aus I unabhängig folgt $\mathbb{1}_I \in \text{STAB}(G)$. Also gilt $\alpha(G, w) = \max\{w^\top \cdot : |x = \mathbb{1}_I, I \text{ unabh.}\} \leq \max\{w^\top \cdot x : x \in \text{STAB}(G)\}$.

($\geq$) Sei $x_{max} \in \text{STAB}(G)$ so, daß $w^\top \cdot x_{max}$ maximal ist. Jedes $x \in \text{STAB}(G)$ hat eine Darstellung $x = \sum_{I \text{ unabh.}} \lambda_I \mathbb{1}_I$ mit $\lambda_I \geq 0$ für alle I. Also $w^\top \cdot x_{max} = \sum_{I \text{ unabh.}} \lambda_I w^\top \cdot \mathbb{1}_I$. Nach Definition von α ist außerdem $w^\top \cdot \mathbb{1}_I \leq \alpha(G, w)$ f. a. I unabhängig und mithin auch $\sum_{I \text{ unabh.}} \lambda_I w^\top \cdot \mathbb{1}_I \leq \alpha(G, w)$. Also ist $w^\top \cdot x_{max} \leq \alpha(G, w)$. □

Folgerung. Das Optimierungsproblem über STAB(G) ist NP-hart.

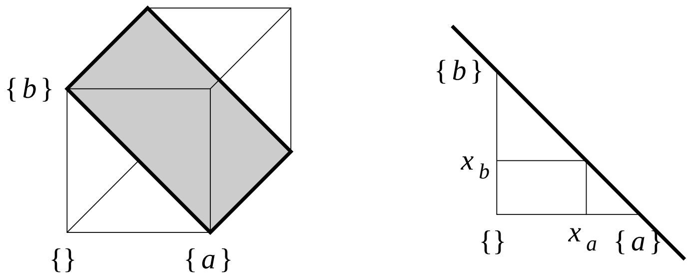
Eine Ungleichung, die für alle $x \in \text{STAB}(G)$ erfüllt ist, heißt *gültige Ungleichung*. Die folgenden Ungleichungen (S1) und (S2) sind, da sie für jeden Vektor $\mathbb{1}_I$, I unabhängig gelten, offenbar gültig.

$$0 \leq x_i \leq 1 \qquad \text{für alle } i \in V \tag{$S1$}$$

$$x_i + x_j \leq 1 \qquad \text{für alle } ij \in E \tag{$S2$}$$

Fortsetzung des Beispiels. Setzt man in S1 und S2 für V und E die Knoten und Kanten aus obigen Beispiel ein, so erhält man folgende Ungleichungen:

$$\begin{aligned} x_a + x_b &\leq 1 \\ x_b + x_c &\leq 1 \\ x_a &\geq 0 \\ x_b &\geq 0 \\ x_c &\geq 0 \end{aligned}$$



Darstellung der Gleichung $x_a + x_b = 1$

Der Durchschnitt der durch diese Ungleichungen induzierten Halbräume ergibt ebenfalls das Polytop STAB(G).

Bemerkung. Es gibt zwei Möglichkeiten, Polytope zu definieren:

- über die konvexe Hülle der Vektoren
- über den Durchschnitt der durch Ungleichungen induzierten Halbräume

Definition. ESTAB$(G) := \{x \in \mathbb{R}^{|V|} : x$ erfüllt die Ungl. (S1) und (S2)$\}$

Da (S1) und (S2) gültige Ungleichungen für STAB(G) sind gilt: STAB$(G) \subseteq$ ESTAB(G). Ausserdem gilt: Die ganzzahligen Vektoren in ESTAB(G) sind gerade die charakteristischen Vektoren von unabhängigen Mengen.

Satz 2. STAB$(G) =$ ESTAB$(G) \iff G$ *bipartit*

Beweis.

($\Longrightarrow$) durch Kontraposition: G nicht bipartit. Dann existiert ein Kreis C der Länge $2k+1$ ($k \geq 1$). Es ist $\frac{1}{2}\mathbb{1}_C \in$ ESTAB(G). Wir behaupten: $\frac{1}{2}\mathbb{1}_C \notin$ STAB(G). Dies sieht man so ein:

Angenommen, $\frac{1}{2}\mathbb{1}_C \in$ STAB(G). Betrachte die Gewichtung $w := \mathbb{1}_C$. Dann ergibt sich folgendes:

(i) Da $|I \cap C| \leq k$, gilt $\mathbb{1}_C^\top \cdot \mathbb{1}_I \leq k$, wenn I unabhängig ist. Für jedes $x \in$ STAB(G) folgt daraus $w^\top \cdot x = \mathbb{1}_C^\top \cdot x = \mathbb{1}_C^\top \cdot \sum_{I \text{ unabh.}} \lambda_I \mathbb{1}_I = \sum_{I \text{ unabh.}} \lambda_I (\mathbb{1}_C^\top \cdot \mathbb{1}_I) \leq k$. Also ist auch $\mathbb{1}_C^\top \cdot \frac{1}{2}\mathbb{1}_C \leq k$.

(ii) $\mathbb{1}_C^\top \cdot \frac{1}{2}\mathbb{1}_C = \frac{1}{2}(2k+1) > k$.

(i) und (ii) führen zum Widerspruch. Damit haben wir für nicht bipartite Graphen STAB$(G) \neq$ ESTAB(G) gezeigt.

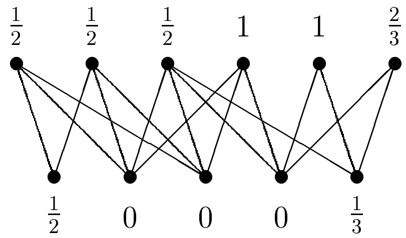
13. Vorlesung, 14.12.95
Mitschrift: Falk Haferkorn

Bemerkung. Ein Polytop P ist die konvexe Hülle einer endlichen Menge von Vektoren. Ein Vektor a eine Ecke von $P \iff (P = \text{conv}(B) \Rightarrow a \in B) \iff \forall\ g$ Gerade durch a gilt: a ist Endpunkt von $g \cap P$. Diese letzte Charakterisierung der Ecken von P und die folgende Übung werden wir im Beweis der Rückrichtung benutzen.

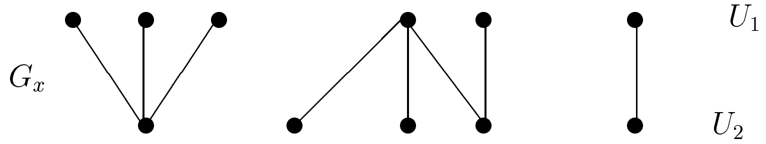
Übung 1. *Seien $P \subseteq Q$ Polytope, wenn jede Ecke von Q in P ist, dann ist $P = Q$.*

Nun zurück zu unserem Beweis. Sei G bipartit und sei $x \in \text{ESTAB}(G)$ ein nicht ganzzahliger Punkt. Wir werden zeigen, es gibt eine Strecke mit Mittelpunkt x die ganz in $\text{ESTAB}(G)$ liegt.

Das folgende Bild zeigt einen bipartiten Graphen mit einem nichtganzzahligen $x \in \text{ESTAB}$, der Wert x_i steht beim Knoten i.



Wir nennen eine Kante $e = (ij)$, für die gilt: $x_i + x_j = 1$ eine *kritische Kante* und betrachten den Subgraphen G_x der kritischen Kanten.



Sei H eine Zusammenhangskomponente von G_x und $i_0 \in V(H)$ mit $0 < x_{i_0} < 1$. Sei U_1, U_2 die Bipartion von H mit $x_{i_0} \in U_1$. Wir definieren $g_t = x - t\mathbb{1}_{U_1} + t\mathbb{1}_{U_2}$. Für welche t ist g_t in ESTAB?

Aus den Bedingungen (S1) folgt: $x_{i_0} - 1 \leq t \leq x_{i_0}$. Aus der Tatsache, daß alle aus H herausragenden Kanten nicht kritisch sind, folgt: $\exists \varepsilon > 0$, so daß: g_t für $-\varepsilon < t < \varepsilon$ die Bedingungen (S2) erfüllt. Damit ist auf der Geraden g_t der Streckenabschnitt $t \in [\max\{-\varepsilon, x_{i_0} - 1\}, \min\{\varepsilon, x_{i_0}\}]$ als ganz in $\text{ESTAB}(G)$

enthalten nachgewiesen. Da $x = g_0$ im inneren dieser Strecke liegt ist x keine Ecke von ESTAB(G). Also sind alle Ecken von ESTAB(G) ganzzahlig und wegen der früher gemachten Beobachtung sind sie charakteristische Vektoren von unabhängigen Mengen, also auch in STAB(G). $\square$

Sei C ein Kreis ungerader Länge $2k + 1$. Für jede unahhängige Menge I gilt $\mathbb{1}_I^T \mathbb{1}_C \leq k$. Daher ist (S3) eine weitere Klasse von gültigen Ungleichungen.

$$\sum_{i \in C \text{Kreis}} x_i \leq \frac{|C| - 1}{2} \tag{S3}$$

Definition. CSTAB(G) $= \{x \in \mathbb{R}^{|V|} : x$ erfüllt (S1),(S2) und (S3) $\}$.

Bemerkung. STAB $\subseteq$ CSTAB $\subseteq$ ESTAB

7.2 Optimierungsprobleme und Polytope

Des *Optimierungsproblem* für ein Polytop P besteht darin zu einem gegebenen Kostenvektor c ein $x \in P$ zu finden, das die Kosten $c^T x$ minimiert.

Ein *Trennungsorakel* für ein Polytop P ist ein Algorithmus, der zu gegebenen x entweder $x \in P$ zusichert oder eine für alle $y \in P$ gültige Ungleichung $u^T y \leq b$ liefert die von x verletzt wird (d.h. $u^T x > b$). Die Hyperebene $H(u, b) = \{z : u^T z = b\}$ trennt P und x.

Satz 3. *P besitzt ein polynomielles Trennungsorakel $\Longrightarrow$ Optimierung über P ist polynomiell.*

Wir skizzieren nun ein polynomielles Trennungsorakel für CSTAB:

Schritt 1: Prüfe alle Ungleichungen aus (S1) und (S2). Ist eine durch x verletzt: gib sie aus, sie ist eine trennende Hyperebene, und Ende.

Schritt 2: Für jede Kante $e = ij$ definiere $c_{ij} = 1 - x_i - x_j$. Wegen (S2) gilt $c_{ij} \geq 0$. Das Gewicht eines Kreises C mit $|C| = l$ bzüglich c_{ij} ist: $\sum_{e \in C} c_e = l - 2 \sum_{i \in C} x_i$. Die Kreisunggleichung für C ist also äquivalent zu $\sum_{e \in C} c_e \geq 1$. Wir müssen also prüfen, ob alle ungeraden Kreise bzgl. der Kosten c_e Gewicht ≥ 1 haben. Ein leichtester ungerader Kreis aber kann, wie die nächsten Übungen zeigt, in polynomieller Zeit gefunden werden.

Übung 2. *Mit $G = (V, E)$ assoziieren wir den bipartiten Graphen $G^* = (V', V'', E^*)$, dessen Knotenmenge aus zwei disjunkten Kopien von V besteht.*

Zu jeder Kante $(u,v) \in E$ enthält E^* die beiden Kanten (u',v'') und (u'',v'), beide mit dem Gewicht von (u,v). Zeige: Leichteste Wege von v' nach v'' in G^* entsprechen den leichtesten ungeraden Kreisen in G die v enthalten.

7.3 Cliquenungleichungen für unabhängige Mengen

Eine weitere Klasse gültiger Ungleichungen für STAB(G):

$$\sum_{i \in Q} x_i \leq 1 \qquad \text{f.a. } Q \text{ Clique} \tag{S4}$$

Definition. QSTAB(G) $= \{x \in \mathbb{R}^{|V|} : x$ erfüllt (S1) und (S4) $\}$.

Da die Kantenungleichungen (S2) in den Cliquenungleichungen (S4) enthalten sind gilt STAB(G) $\subseteq$ QSTAB(G) $\subseteq$ ESTAB(G).

Definition. Lösungen des Optimierungsproblems über QSTAB heißen *fraktionale gewichtete Unabhängigkeitszahlen*

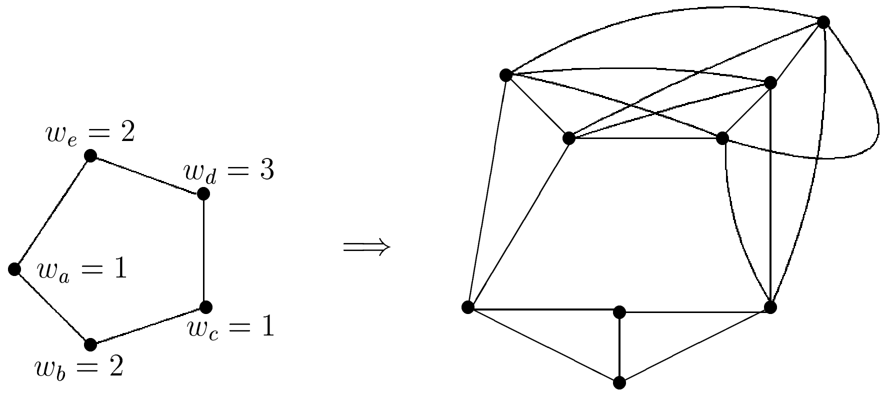
$$\alpha^*(G,w) = \max_{x \in \text{QSTAB}} w^T x$$

Für $w : V \longrightarrow \mathbb{N}$ sei die gewichtete chromatische Zahl definiert als

$$\chi(G,w) = \min(\sum_I a_I \ : \ a_I \in \mathbb{N} \text{ und } \sum_{I, i \in I} a_I = w_i)$$

Zu gegebenen w sei G_w der Graph, der aus G entsteht, wenn man Knoten i durch einen vollständigen Graphen K_{w_i} ersetzt (diese Konstruktion wird im folgenden Bild dargestellt). Es gilt $\chi(G,w) = \chi(G_w)$.

Bsp.:



Die Partition $\big\{\{a,d_1\},\{b_1,d_2\},\{b_2,d_3\},\{c,e_1\},\{e_2\}\big\}$ ist eine optimale Färbung von G_w. Die gewichtete chromatische Zahl $\chi(G,w)$ erhält man durch die zur Färbung von G_w gehörige Belegung: $a_{\{a,d\}}=1$, $a_{\{b,d\}}=2$, $a_{\{c,e\}}=1$, $a_{\{e\}}=1$.

Definition. Wir definieren $\kappa(G,w)=\chi(\overline{G},w)$

Übung 3. *Wie sieht der Graph G^w aus, für den $\kappa(G,w)=\kappa(G^w)$ gilt?*

Satz 4. $\alpha(G,w)\le\alpha^*(G,w)\le\kappa(G,w)$

Beweis. Die linke Ungleichung folgt sofort aus $\mathrm{STAB}(G)\subseteq\mathrm{QSTAB}(G)$.

Sei $Q_1,Q_2,...,Q_k$ eine optimale gewichtete Cliquenüberdeckung, d.h. $k=\kappa(G,w)$ und jeder Knoten v kommt in w_v der Cliquen vor. Die zweite Eigenschaft können wir auch so schreiben:

$$w=\mathbb{1}_{Q_1}+\mathbb{1}_{Q_2}+..+\mathbb{1}_{Q_k}$$

Für $x\in\mathrm{QSTAB}(G)$ gilt also

$$w^Tx \;=\; \sum_{j=1}^{k}\mathbb{1}_{Q_j}^Tx \;\le\; \sum_{j=1}^{k}1 \;=\; \kappa(G,w).$$

Damit ist $\alpha^*(G,w)=\max_{x\in\mathrm{QSTAB}}w^Tx\le\kappa(G,w)$ gezeigt. □

7.4 Das Perfekt Graph Theorem

14. Vorlesung, 19.12.95
Mitschrift: Helmut Weil

Satz 5. *Für einen Graphen $G=(V,E)$ ist äquivalent:*

(*i*) *G ist χ-perfekt*

(*ii*) $\chi(G,w)=\omega(G,w)$ *für alle $w:V\to\mathbb{N}$*

(*iii*) $\mathrm{STAB}(G)=\mathrm{QSTAB}(G)$

(*iv*) *G ist α-perfekt*

(*v*) $\alpha(G,w)=\kappa(G,w)$ *für alle $w:V\to\mathbb{N}$*

(*vi*) $\mathrm{STAB}(\overline{G})=\mathrm{QSTAB}(\overline{G})$

Beweis. $(i)\Rightarrow(ii)$ zeigen wir mit Induktion nach $w(V)=\sum_{v\in V}w(v)$.

Wenn $w \leq \mathbb{1}_V$, dann gibt es ein $A \subseteq V$ so daß $w = \mathbb{1}_A$ und es ist $G_w = G[A]$ und damit (und wegen (i))

$$\chi(G, w) = \chi(G[A]) = \omega(G[A]) = \omega(G, w).$$

Wenn $w > \mathbb{1}_V$ gibt es ein $i_0 \in V$ mit $w(i_0) > 1$. Setze $w' = w - \mathbb{1}_{\{i_0\}}$. Nach Induktionsannahme gibt es (nicht notwendig verschiedene) unabhängige Mengen $I_1, I_2, \ldots, I_k$, so daß jeder Knoten v in $w'(v)$ dieser Mengen enthalten ist und $k = \chi(G, w') = \omega(G, w')$. Sei $S \in \{I_1, \ldots, I_k\}$, so daß $i_0 \in S$. Setze $w'' = w - \mathbb{1}_S$. Wir behaupten: $\omega(G, w'') \leq \omega(G, w) - 1$. Um das zu sehen, betrachten wir für eine Clique Q mit $w''(Q) = \omega(G, w'')$ die folgenden beiden Fälle: 1. Fall: $S \cap Q \neq \emptyset$, dann gilt $\omega(G, w'') = w''(Q)$ (nach Wahl von Q) und $w''(Q) = w(Q) - 1$ (weil $S \cap Q \neq \emptyset$) und $w(Q) - 1 \leq \omega(G, w) - 1$. 2. Fall: $S \cap Q = \emptyset$, dann gilt $\forall i \in Q : w''(i) = w'(i) = w(i)$ und es folgt

$$\omega(G, w'') = w''(Q) = \sum_{i \in Q} w''(i) = \sum_{i \in Q} w'(i) = \sum_{i \in Q} \sum_{j=1}^{k} \delta_{i \in I_j} = \sum_{j=1}^{k} |Q \cap I_j|$$

Weil $|Q \cap I_j| \leq 1$ (für $j = 1, \ldots, k$) und $|Q \cap I_j| = 0$ für $I_j = S$, folgt

$$\omega(G, w'') \leq k - 1 = \omega(G, w') - 1 \leq \omega(G, w) - 1$$

Damit ist die Behauptung gezeigt und wir können im Beweis des Satzes fortfahren. Die Induktionsannahme für w'' liefert uns unabhängige Mengen $J_1, \ldots, J_{k''}$, so daß jeder Knoten v in $w''(v)$ davon enthalten ist und $k'' = \chi(G, w'') = \omega(G, w'')$. Die unabhängigen Mengen $J_1, \ldots, J_{k''}, S$ sind eine Färbung zur Gewichtung w. Diese Färbung ist optimal, weil

$$\chi(G, w) \leq k'' + 1 = \chi(G, w'') + 1 = \omega(G, w'') + 1 \leq \omega(G, w)$$

und trivialerweise $\omega(G, w) \leq \chi(G, w)$ gilt. Also ist $\omega(G, w) = \chi(G, w)$.

$(ii) \Rightarrow (iii)$. $\mathrm{STAB}(G) \subseteq \mathrm{QSTAB}(G)$ ist bekannt. Umgekehrt genügt es für *rationale* $x \in \mathrm{QSTAB}(G)$ zu zeigen, daß $x \in \mathrm{STAB}(G)$ gilt. Sei also $x \in \mathrm{QSTAB}(G)$ rational, dann gibt es ein q, so daß $qx \in \mathbb{N}^V$. Die Ungleichungen $(S4)$ sind definitionsgemäß für x erfüllt, d.h. $\sum_{i \in Q} x_i \leq 1$ für alle Cliquen Q. Es folgt $\omega(G, qx) = \max\limits_{Q} \sum\limits_{i \in Q} qx_i \leq q$ und wegen (ii) auch $\chi(G, qx) \leq q$. Sei $I_1, \ldots, I_k$ eine optimale qx-Färbung von G (also $k \leq q$), dann gilt

$$qx = \sum_{i=1}^{k} \mathbb{1}_{I_i} \text{ und damit } x = \sum_{i=1}^{k} \frac{1}{q} \mathbb{1}_{I_i}$$

Weil $(1-\frac{k}{q}), \frac{1}{q} \geq 0$ und $(1-\frac{k}{q})+\sum_{i=1}^{k}\frac{1}{q}=1$, ist x also als Konvexkombination der unabhängigen Mengen $\emptyset, I_1, \ldots, I_k$ darstellbar. Damit ist $x \in \mathrm{STAB}(G)$ gezeigt.

$(iii) \Rightarrow (iv)$. Aus $\mathrm{STAB}(G) = \mathrm{QSTAB}(G)$ folgt durch Projektion auf die zu $W \subseteq V$ gehörenden Koordinaten, daß $\mathrm{STAB}(G[W]) = \mathrm{QSTAB}(G[W])$. Zum Nachweis der α-Perfektheit (d.h. $\alpha(G[W]) = \kappa(G[W])$ für alle $W \subseteq V$) genügt es also $\alpha(G) = \kappa(G)$ zu zeigen.

Sei F die Seite von $\mathrm{STAB}(G)$, auf der alle Vektoren $\mathbb{1}_I$ für unabhängige Mengen I mit $|I| = \alpha(G)$ liegen. Wegen (iii) gibt es eine Cliquenungleichung, die von allen Vektoren in F mit Gleichheit erfüllt wird, d.h. es gibt eine Clique Q, so daß $\mathbb{1}_Q^\mathsf{T} x = 1$ für alle $x \in F$. Das bedeutet, daß jede unabhängige Menge der Größe $\alpha(G)$ ein Element mit Q gemeinsam hat. Für $G' = G[V - V(Q)]$ gilt also $\alpha(G') = \alpha(G) - 1$. Wenn wir induktiv annehmen, daß $\alpha(G') = \kappa(G')$, d.h. es gibt eine Cliquenüberdeckung $Q_1, \ldots, Q_k$ von G' mit $k = \alpha(G')$. Dann ist $Q_1, \ldots, Q_k, Q$ eine Cliquenüberdeckung von G mit $\alpha(G') + 1 = \alpha(G)$ Cliquen.

Die Implikationen $(iv) \Rightarrow (v)$, $(v) \Rightarrow (vi)$ und $(vi) \Rightarrow (i)$ ergeben sich durch Vertauschen von G und $\overline{G}$. □

Satz 6. *Das Optimierungsproblem für* QSTAB *ist NP-hart.*

Beweisskizze. Eine konvexe Menge $P \subseteq \mathbb{R}_+^n$ heißt *konvexe Ecke*, wenn aus $0 \leq x \leq y$ und $y \in P$ folgt, daß $x \in P$. Der *Antiblocker* von P ist

$$\mathrm{ABL}(P) = \{x \in \mathbb{R}_+^n : y^\mathsf{T} x \leq \mathbb{1} \text{ für alle } y \in P\}$$

Übung 4.

- $\mathrm{ABL}(\mathrm{ABL}(P)) = P \iff P$ *ist konvexe Ecke.*
- STAB *und* QSTAB *sind konvexe Ecken.*

Wegen $x^\mathsf{T}\mathbb{1}_I \leq 1 \Rightarrow x^\mathsf{T} y \leq 1$ für alle $y \in \mathrm{STAB}(G)$ gilt:

$$\mathrm{ABL}(\mathrm{STAB}(G)) = \{x : x^\mathsf{T}\mathbb{1}_I \leq 1 \text{ für alle } I \text{ unabh. in } G\}$$
$$= \{x : x^\mathsf{T}\mathbb{1}_Q \leq 1 \text{ für alle } Q \text{ Clique in } \overline{G}\} = \mathrm{QSTAB}(\overline{G})$$

Übung 5. *Ein Algorithmus, der das Problem* $\max\{w^\mathsf{T} x | x \in P\}$ *löst, kann als Trennungsorakel für das Optimierungsproblem über* $\mathrm{ABL}(P)$ *verwendet werden.*

Wenn wir jetzt annehmen, wir können über QSTAB(G) in polynomieller Zeit optimieren, dann liefert uns das ein polynomielles Trennungsorakel über

$$\mathrm{ABL}(\mathrm{QSTAB}(G)) = \mathrm{ABL}(\mathrm{ABL}(\mathrm{STAB}(\overline{G}))) = \mathrm{STAB}(\overline{G})$$

und das führt zu einem Widerspruch da Optimieren über STAB($\overline{G}$) NP-hart ist. $\triangle$

Wir haben gesehen, daß Optimieren sowohl über STAB als auch über QSTAB NP-hart ist. Es läßt sich allerdings zeigen, daß es für jeden Graphen G eine konvexe Ecke TH(G) gibt, die i.A. kein Polytop ist, so daß STAB(G) $\subseteq$ TH(G) $\subseteq$ QSTAB(G) und Optimierung über TH(G) in polynomieller Zeit möglich ist. Der oben bewiesene Satz zeigt, daß für perfekte Graphen gilt:

$$\mathrm{STAB}(G) \;=\; \mathrm{TH}(G) \;=\; \mathrm{QSTAB}(G)$$

Folgerung. Die Unabhängigkeitszahl $\alpha(G)$ eines perfekten Graphen G kann in polynomieller Zeit bestimmt werden.

7.5 Die Lovász Ecke TH(G)

Definition. Eine *orthonormale Repräsentation* eines Graphen $G = (V, E)$ besteht aus einem Vektor $u_i \in \mathbb{R}^N$ für jedes $i \in V$, so daß gilt (i) $\|u_i\| = 1$ für alle $i \in V$ und (ii) $u_i^{\mathsf{T}} u_j = 0$ für alle $ij \notin E$.

Bemerkung. Sei $\phi : V \to \{1..k\}$ eine Färbung von $\overline{G}$ und $\epsilon_1..\epsilon_k$ eine Menge paarweise orthogonaler Einheitsvektoren, dann ist $i \to \epsilon_{\phi(i)}$ eine orthogonale Repräsentation von $\overline{G}$. Insbesondere gibt es also orthogonale Repräsentationen von G in allen Dimensionen $\geq \kappa(G)$.

Zu einer orthonormale Repräsentation $\{u_i\}_{i \in V}$ von G und einem $c \in \mathbb{R}^N$ mit $\|c\| = 1$ definieren wir den Vektor $\xi \in \mathbb{R}^n$, $n = |V|$ durch

$$\xi_i = (c^{\mathsf{T}} u_i)^2 \qquad (\star)$$

†Abschnitt 7.5 wurde in der Vorlesung nicht mehr behandelt — Weihnachtsferien!

Lemma 7. *Die Ungleichungen* $\xi^\mathsf{T} x \leq 1$ *sind gültig für* STAB(G).

Beweis. Sei $\{v_a\}$ eine Orthonormalbasis des $\mathbb{R}^N$. Der Koeffizient von v_a in der Entwicklung von c zu dieser Basis ist $c^\mathsf{T} v_a$. Also ist $1 = \|c\|^2 = \sum_a (c^\mathsf{T} v_a)^2$.

Ist I unabhängig, dann kann $\{u_i\}_{i \in V}$ zu einer Orthonormalbasis des $\mathbb{R}^N$ erweitert werden. Daher gilt für jedes I

$$\xi^\mathsf{T} \mathbb{1}_I = \sum_{i \in I} \xi_i = \sum_{i \in I} (c^\mathsf{T} u_i)^2 \leq 1.$$

Wie üblich überträgt sich die Ungleichung auf alle $x \in$ STAB(G). □

Wir verwenden diese Ungleichungen zur Definition der Lovász Ecke TH(G).

$$\mathrm{TH}(G) = \{x \in \mathbb{R}_+^V : \xi^\mathsf{T} x \leq 1 \text{ für alle } \xi = \xi(\{u_i\}, c) \text{ wie in } \star\}$$

Sei Q eine Clique und seien $\{c\} \cup \{u_i\}_{i \in V \setminus Q}$ paarweise orthogonal. Definiert man $u_j = c$ für $j \in Q$, dann ist die zu $\{u_i\}_{i \in V}$ und c gehörige Ungleichung $\xi^\mathsf{T} x \leq 1$ gerade die Clique-Ungleichung $\sum_{i \in Q} x_i \leq 1$. Also ist TH($G$) $\subseteq$ QSTAB(G).

Als direkte Folgerung aus den angestellten Überlegungen erhalten wir.

Satz 8. *Für alle Graphen gilt* STAB(G) $\subseteq$ TH(G) $\subseteq$ QSTAB(G). *Für perfekte Graphen gilt* STAB(G) = TH(G) = QSTAB(G).

Die Lovász Theta Funktion

Wir definieren die *Lovász Theta Funktion* $\vartheta : \mathbb{R}_+^n \to \mathbb{R}$ durch die Lösungen des Optimierungsproblems über TH(G),

$$\vartheta(G, w) = \max\{w^\mathsf{T} x : x \in \mathrm{TH}(G)\}.$$

Der nächste Satz, das *Sandwich Theorem*, ist eine direkte Folgerung aus den im letzten Abschnitt angestellten Überlegungen. Es sei $\vartheta(G) = \vartheta(G, \mathbb{1})$.

Satz 9. *Für jeden Graphen G gilt* $\alpha(G, w) \leq \vartheta(G, w) \leq \alpha^*(G, w) \leq \kappa(G, w)$ *und speziell* $\alpha(G) \leq \vartheta(G) \leq \kappa(G)$. □

Zum Kapitel

Der in diesem Kapitel vorgestellte Zugang zum Problem eine maximum unabhängige Menge oder doch zumindest ihren Wert zu bestimmen stammt aus der polyedrischen Optimierung. Als Referenz zu diesem Thema sei die umfassende Monographie von Grötschel, Lovász und Schrijver [22] genannt. Kapitel 9 in diesem Buch beschäftigt sich mit *Stable sets in graphs*. Tatsächlich laßen sich maximum unabhängige Mengen und eine optimale Färbungen in perfekten Graphen in polynomieller Zeit bestimmen [22].

Details zur Optimierbarkeit von $\mathrm{TH}(G)$ findet man in [22] und in einem ergänzenden Artikel von Knuth [28]. In diesen Artikeln werden 5 überaschend verschieden Möglichkeiten die Funktion $\vartheta(G, w)$ zu definieren vorgestellt und als äquivalent nachgewiesen. Die Methoden zur Approximation der Unabhängigkeitszahl durch ϑ wurden verallgemeinert. Mit dem Namen *semidefinite Programmierung* versehen, konnten sie inzwischen auf verschiedene Probleme angewandt werden. Einen Überblick geben Vandenberghe und Boyd [50].

Weitere nichtlineare Methoden zur Gewinnung von Schranken für die Unabhängigkeitszahl wurden von Lovász [32] vorgestellt.

8 Minimal imperfekte Graphen

15. Vorlesung, 9. Januar 96
Mitschrift: Clemens Hendler

Definition. Ein Graph G heißt *minimal imperfekt (Mip)* falls gilt:
G nicht perfekt und fa. $v \in V$ ist $G[V - v]$ perfekt.

Bemerkung. G Mip, I unabh. in $G \Rightarrow \omega(G[V - I]) = \omega(G)$

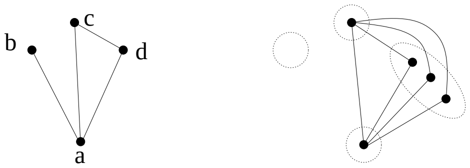
Beweis. Angenommen $\omega(G[V - I]) < \omega(G)$
$G[V - I]$ perfekt $\Rightarrow$ ex. Färbung von $G[V - I]$ mit $\omega(G[V - I])$ Farben. Diese Färbung ist mit einer zusätzlichen Farbe für I zu einer Färbung von G erweiterbar. Also gilt: $\chi(G) \leq \chi(G[V - I]) + 1 = \omega(G[V - I]) + 1 \leq \omega(G)$. Also mit $\omega(G) \leq \chi(G) \Rightarrow \omega(G) = \chi(G)$. Da G Mip gilt fa $W \subset V : G[W]$ perfekt, also auch G — Widerspruch. □

Bemerkung. Ungerade Kreise und ihre Komplemente sind Mip. (Die Vermutung von Bergebesagt ja gerade, daß ungeraden Kreise und ihre Komplemente genau die Mip-Graphen sind).

Bemerkung. G Mip $\Leftrightarrow G^c$ Mip. (Das folgt direkt aus dem Perfekt Graph Theorem).

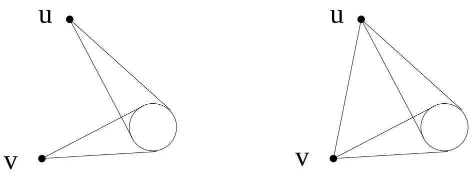
Für einen Graphen $G = (V, E)$ und eine Abb. $w : V \to \mathbb{N}$ bezeichne $G \circ w$ (Multiplikation von G mit w) den Graphen den man erhält, indem man $v \in V$ durch unabh. Menge der Größe $w(v)$ ersetzt.

Beispiel. $w(a, b, c, d) = (1, 0, 1, 3)$



Übung 1. *Aus G perfekt und* $w : V \to \mathbb{N}$ *folgt* $G \circ w$ *perfekt*

Gilt für zwei Knoten $u \neq v$ $N(u) = N(v)$, so nennt man sie *Zwillinge*, und im Fall $N(u) \cup \{u\} = N(v) \cup \{v\}$ *Cozwillinge.*



Zwilling CoZwilling

Bemerkung. G Mip $\Rightarrow$ G enthält keine Zwillinge und keine Cozwillinge.

Beweis. In Hinblick auf die vorige Bemerkung beschränken wir uns auf den Fall der Zwillinge. Aus dem Perfekt Graph Theorem (PGT) folgt: Ist G perfekt, $w : V \to \mathbb{N}$, dann ist auch $G \circ w$ perfekt.
Annahme: ex Mip G mit Zwillingen u und v. Da G Mip ist $G[V \setminus v]$ perfekt. Sei $w : V \to \mathbb{N}$ mit $w(u) = 2$ und fa. $s \in V \setminus \{u, v\} : w(s) = 1$. Dann ist $G[V \setminus v] \circ w = G$, also G perfekt — Widerspruch □

8.1 Eine Charakterisierung perfekter Graphen

Satz 1. [Lovász] *G perfekt* $\Leftrightarrow$ $f a.W \subseteq V : \omega(G[W]) \cdot \alpha(G[W]) \geq |W|$

Übung 2. *Mit dem Satz gilt: G ist* χ*-perfekt* $\Leftrightarrow$ *G ist* α*-perfekt.*

Das heißt also, daß der zu beweisende Satz gleichwertig zu (PGT) ist (vorausgesetzt wir benötigen (PGT) nicht für den Beweis), was uns schon ahnen läßt, daß der Beweis kein Dreizeiler wird. Im nachfolgenden Beweis benutzen wir (PGT) der Einfachheit halber an einer Stelle, es sei aber erwähnt, daß sich dies durchaus vermeiden ließe.

Beweis. ($\Rightarrow$) fa. $G : \chi(G) \cdot \alpha(G) \geq |V(G)|$ (zerlege Graphen in $\chi(G)$ unabh. Mengen, jede ist $\leq \alpha(G)$). Für perfekte Graphen gilt zusätzlich: fa. $W \subseteq V$: $\chi(G[W]) = \omega(G[W])$.
($\Leftarrow$) Angenommen, G nicht perfekt. Sei G ein bzgl. $|V|$ minimales Gegenbeispiel. Dann gilt: G ist Mip (wäre G kein Mip könnten wir einen nichtperfekten Subgraphen betrachten. Der erfüllt unsere Prämisse weil G sie erfüllt und ist also auch ein Gegenbeispiel). Wir zeigen zunächst folgende Hilfsbehauptung:
Behauptung. $\omega(G \circ w) \cdot \alpha(G \circ w) \geq |V(G \circ w)| = \sum_{i \in V} w(i)$ ($\clubsuit$)
Bew: Annahme: ex. $w : V \to \mathbb{N}$ mit $\omega(G \circ w) \cdot \alpha(G \circ w) < |V(G \circ w)|$. Sei w eine solche Abb., die $\sum_{i \in V} w(i)$ minimiert. Da nach Voraussetzung die rechte Seite des Satzes, also die Ungleichung für G und seine Subgraphen erfüllt ist, ex. $i \in V$ mit $w(i) > 1$. Sei i_0 so gewählt und $w_0 := w(i_0)$. Betrachte nun $w' := w - \mathbb{1}_{\{i_0\}}$. Wegen der Minimalität von w gilt:

$$\omega(G \circ w') \cdot \alpha(G \circ w') \geq \sum_{i \in V} w'(i) = \sum_{i \in V} w(i) - 1,$$

und nach Annahme:

$$\omega(G \circ w) \cdot \alpha(G \circ w) \leq \sum_{i \in V} w(i) - 1.$$

Da $G \circ w'$ Subgraph von $G \circ w$, gilt:

$$\omega(G \circ w') \leq \omega(G \circ w) \text{ und } \alpha(G \circ w') \leq \alpha(G \circ w);$$

somit muß insgesamt gelten:

$$\omega(G \circ w') = \omega(G \circ w) =: p, \quad \alpha(G \circ w') = \alpha(G \circ w) =: r,$$

also $\sum_{i \in V} w(i) = p \cdot r + 1$.

Sei nun $w'' = w - w_0 \cdot \mathbb{1}_{\{i_0\}}$. Dann entsteht $G \circ w''$ aus $G \circ w$ durch Entfernen aller Kopien von i_0, bzw durch Multiplikation aus $G[V - i_0]$. Als Subgraph des Mip G ist $G[V - i_0]$ perfekt, also auch $G \circ w''$ (PGT) und es gilt: $\kappa(G \circ w'') = \alpha(G \circ w'') \leq \alpha(G \circ w) = r$. Es ex. also eine Cliquenüberdeckung von $G \circ w''$ bestehend aus r Cliquen. Seien $C_1, ..., C_r$ so gewählt, obdA gelte $|C_1| \geq ... \geq |C_r|$. Es gilt:

- $|C_i| \leq p$ fa. $i = 1..r$
- $\sum_{i=1..r} |C_i| = \sum_{i \in V} w''_i = (\sum_{i \in V} w_i) - w_0 = p \cdot r + 1 - w_0$

$\Rightarrow$ für höchstens $w_0 - 1$ der C_i gilt $|C_i| < p$.
$\Rightarrow |C_1| = ... = |C_{r-(w_0-1)}| = p$. ($r - (w_0 - 1) \geq 1$, da $r = \alpha(G \circ w)$ und w_0 Größe der unabh. Menge der Kopien von i_0 in $G \circ w$).

Sei $W := \{i_0\} \cup \bigcup_{i=1..r-w_0+1} C_i$ und $H := G \circ w[W]$, also $H = G \circ w'''$ für ein geeignetes $w''' < w$ (komponentenweise $\leq$, für zumindest eine $<$). Da w minimal gewählt war ist $\omega(H) \cdot \alpha(H) \geq |V(H)| = p \cdot (r - w_0 + 1) + 1$. Wegen $\omega(H) \leq \omega(G \circ w) = p$ gilt $\alpha(H) \geq r - w_0 + 2$.

Sei I unabh. in H mit $|I| = \alpha(H)$. Dann gilt: $i_0 \in I$ weil $W - i_0$ Vereinigung von $r - w_0 + 1$ Cliquen ist. Sei I'maximale unabh. Menge in $G \circ w$, die I enthält. Mit i_0 sind auch sämtliche Kopien von i_0 in I', also gilt $|I'| \geq |I| + w_0 - 1 \geq r - w_0 + 2 + w_0 - 1 = r + 1 > \alpha(G \circ w)$ — Widerspruch. damit ist ($\clubsuit$) bewiesen.

Da G Mip gilt: fa. I unabh. in $G : \omega(G-I) = \chi(G-I) \geq \chi(G) - 1 \geq \omega(G)$, also $(\dagger) : \omega(G - I) = \omega(G)$. Sei $\mathcal{I} := \{I \subseteq V : I$ unabhängig in $G\}$. Für jedes $I \in \mathcal{I}$ wähle eine Clique C_I mit

- $I \cap C_I = \emptyset$
- $|C_I| = \omega(G)$

(möglich wegen $(\dagger)$). Konstruiere $w : V \to \mathbb{N}$ mit $w(v) := |\{I \in \mathcal{I} : v \in C_I\}|$. Dann gilt:

(i) $|V(G \circ w)| = \sum_{v \in V} w(v) = \sum_{v \in V} \sum_{I \in \mathcal{I}} \delta_{v \in C_I} = |\mathcal{I}| \cdot \omega(G)$

(ii) $\omega(G \circ w) \leq \omega(G)$

(iii) $\alpha(G \circ w) = \max_{I \in \mathcal{I}} \sum_{v \in I} w(v) = \max_{I \in \mathcal{I}} \sum_{v \in I} \sum_{J \in \mathcal{I}} \delta_{v \in C_J} = \max_{I \in \mathcal{I}} \sum_{J \in \mathcal{I}} |C_J \cap I|$.

Da J unabhängig und C_I Clique ist $|C_J \cap I| \leq 1$. Für $I = J$ gilt $|C_J \cap I| = 0$, also $\alpha(G \circ w) \leq |\mathcal{I}| - 1$. Insgesamt gilt nun:
$\omega(G \circ w) \cdot \alpha(G \circ w) \leq \omega(G) \cdot (|\mathcal{I}| - 1) < |V(G \circ w)|$ – Widerspruch zu ($\clubsuit$) □

Folgerung. G Mip $\Rightarrow \omega(G) \cdot \alpha(G) = |V| - 1$

Beweis. Es gilt:

$$\begin{aligned} \omega(G[V-v]) \cdot \alpha(G[V-v]) &\geq |V| - 1 \\ \omega(G) \cdot \alpha(G) &\leq |V| - 1 \\ \omega(G[V-v]) &\leq \omega(G) \\ \alpha(G[V-v]) &\leq \alpha(G) \end{aligned}$$

also gilt in allen Ungleichungen die Gleichheit. □

8.2 (ω, α)-zerlegbare Graphen

16. Vorlesung, 11.1.96
Mitschrift: Carsten Ihlemann

Definition. Sei $G = (V, E)$ Graph, $p, r \in \mathbb{N}, r > 1$. G heißt (p, r)-zerlegbar falls $|V(G)| = pr + 1$ und für alle $v \in V$ gilt: $\chi(G - v) = p$ und $\kappa(G - v) = r$.

Lemma 2. *Sei G (p, r)-zerlegbar. Dann ist $\omega(G) = p$ und $\alpha(G) = r$.*

Beweis. Eine minimale Clique-Überdeckung von $G - v$ teilt $G - v$ in r Cliquen. Da $|G - v| = p \cdot r$ ist, muß eine Clique Größe mindestens p haben. Also $\omega(G) \geq \omega(G - v) \geq p$. Eine p-Färbung von $G - v$ partitioniert $G - v$ in p unabhängige Mengen, von denen eine mindestes die Größe r haben muß. Also $\alpha(G) \geq \alpha(G - v) \geq r$.

Sei andereseits C eine maximale Clique in G und $v \notin C$. Das geht, da ein (p, r)-zerlegbarer Graph nicht vollständig ist. C ist dann auch maximale Clique in $G - v$. Dann folgt: $p = \kappa(G - v) \geq \omega(G - v) = \omega(G)$. Also: $\omega(G) = p$.
α analog. $\square$

Übung 3. *Zeige $\{G$: G Mip $\} \subset \{G$: G pr-zerlegbar $\} \subset \{G$: G imperfekt $\}$.*

Satz 3. *G sei (ω, α)- zerlegbar mit $|V(G)| = n$. Dann gibt es n verschiedene Cliquen $C_1, .., C_n$ der Größe ω und n verschiedene unabhängige Mengen $I_1, .., I_n$ der Größe α so daß*

(1) *jedes $v \in V$ kommt in ω vielen der C_i vor*

(2) *jedes $v \in V$ kommt in α vielen der I_j vor*

(3) *$C_i \cap I_j = \emptyset \Leftrightarrow i = j$.*

Beweis. Sei C eine maximum Clique. Also $|C| = \omega$. Für jedes $v \in C$ sei $\mathcal{C}_v$ eine minimale Cliquen-Überdeckung von $G - v$. Nach Lemma 5 besteht $\mathcal{C}_v$ aus α Cliquen der Grösse $\omega(G)$. Sei $\mathcal{C} := \{C\} \cup \bigcup_{v \in C} \mathcal{C}_v = \{C_1, ..C_{\alpha\omega+1}\}$.

Behauptung: Jeder Knoten ist durch $\mathcal{C}$ ω-fach überdeckt, d.h $\mathcal{C}$ erfüllt (1). Beweis: Jedes $u \notin C$ kommt in jeder Überdeckung $\mathcal{C}_v$, $v \in C$ genau einmal vor, also $|C| = \omega$ mal. Ist $u \in C$, dann wird u von C und von jedem $\mathcal{C}_v$ mit $v \neq u$ überdeckt.

Sei A die $n \times n$-Matrix, deren i-te Zeile der Vektor $\mathbb{1}_{C_i}^\top$ ist. Nach dem eben gezeigten gilt: $\mathbb{1}^\top A = \omega \mathbb{1}^\top$. Für jedes i wähle $v_i \in C$ beliebig. Sei $\mathcal{I}_i$

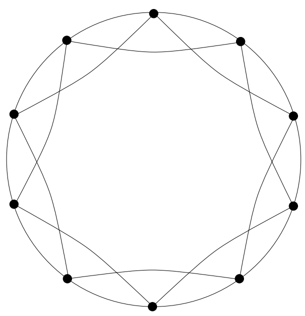
eine minimale Überdeckung von $G - v_i$ durch unabhängige Mengen (d.h. eine optimale Färbung von $G - v_i$), dann ist $|\mathcal{I}_i| = \omega$ und jedes $I \in \mathcal{I}_i$ hat Größe α. Sei I_i die eindeutig bestimmte unabhängige Menge aus $\mathcal{I}_i$, für die $I_i \cap C_i = \emptyset$ gilt. Sei B die $n \times n$ Matrix deren i-te Zeile der Vektor $\mathbb{1}_{I_i}^\top$ ist. Nun gilt $B \cdot \mathbb{1} = \alpha \cdot \mathbb{1}$ also $\mathbb{1}^\top A \cdot B^\top = \omega \mathbb{1}^\top B^\top = \omega\alpha\mathbb{1}^\top = (n-1)\mathbb{1}^\top$. Weil $a_i \cdot b_i^\top = 0$ nach der Wahl von $b_i = \mathbb{1}_{I_i}$, besteht die Diagonale von $A \cdot B^\top$ nur aus Nullen. Also ist $A \cdot B^\top = J - I$, wobei J die überall 1 Matrix ist. Damit ist (3) für die Cliquen C_i und die unabhängigen Mengen I_i gezeigt. Insbesondere sind die Cliquen/unabhängige Mengen jeweils untereinander verschieden.

Die Matrix $J - I$ ist nicht-singulär also sind auch A und B nicht-singulär. Aus $\mathbb{1}^\top A = \omega \mathbb{1}^\top$ folgt nun $\mathbb{1}^\top A^{-1} = \frac{1}{\omega}\mathbb{1}^\top$. Daher gilt $\mathbb{1}^\top B = \mathbb{1}^\top B \cdot A^\top \cdot (A^\top)^{-1} = \mathbb{1}^\top \cdot (J - I) \cdot (A^\top)^{-1} = (n-1) \cdot \mathbb{1}^\top \cdot (A^\top)^{-1} = \frac{(n-1)}{\omega} \cdot \mathbb{1}^\top = \alpha \cdot \mathbb{1}^\top$. Die Gleichung $\mathbb{1}^\top \cdot B = \alpha \cdot \mathbb{1}^\top$ entspricht der noch austehenden Behauptung (2). □

Übung 4. *Beweise, daß $J - I$ nicht-singulär ist. (Hinweis: finde n verschieden Eigenvektoren).*

Definition. $C_{\omega\alpha+1}^{\omega-1}$ sei der Graph auf $V = \{v_0, ..., v_{\omega\alpha}\}$ und der Kantenmenge
$$\Big\{\{v_i, v_j\} : i \neq j, ; (i - j \bmod \omega\alpha + 1) \leq \omega - 1\Big\}.$$

Beispiel. C_{10}^2



Bemerkung. $C_{2\alpha+1}^1$ ist ungerader Kreis und $C_{2\omega+1}^{\omega-1}$ ist Komplement eines ungeraden Kreises.

Übung 5. *Für alle $\omega, \alpha \geq 2$ ist $C_{\omega\alpha+1}^{\omega-1}$ ein (ω,α)-zerlegbarer Graph.*

Satz 4. *Für $\omega, \alpha \geq 3$ ist $C_{\omega\alpha+1}^{\omega-1}$ kein Mip.*

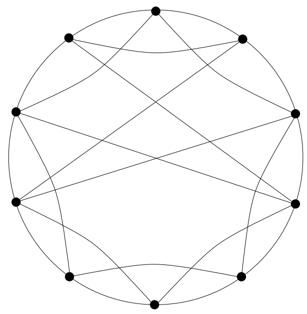
Beweis. Sei $G = C_{\omega\alpha+1}^{\omega-1}$ und $S = \{v_1, v_\omega, v_{\omega+2}, v_{2\omega+1}, v_{(\alpha-1)\cdot\omega+1}, v_{\alpha\omega}\}$ Behauptung: $G' := G[V - S]$ ist nicht-perfekt. Es ist $\omega(G') \leq \omega - 1$ weil S aus jeder ω-Clique von G, d.h aus jedem ω-Abschnitt, einen Knoten enthält.

Annahme: ex. Färbung mit $\omega - 1$ Farben. Sei $R = \{1, .., \omega - 2\}$ und sei F eine Färbung von G mit den Farben $\{\text{rot}\} \cup R$. OBdA sei $F(v_0) =$ rot. Weil $v_0, v_2, v_3, .., v_{\omega-1}$ eine Clique in G' ist gilt: $F(v_2, .., v_{\omega-1}) = R$. Weil $v_2, v_3, .., v_{\omega-1}, v_{\omega+1}$ eine Clique in G' ist gilt: $F(v_{\omega+1}) =$ rot. Aus der Betrachtung weiterer Cliquen erhält man: $F(v_{\omega+3}, .., v_{2\omega}) = R \Rightarrow F(v_{2\omega+2}) =$ rot $\Rightarrow F(v_{2\omega+3}, .., v_{3\omega}) = R \ldots \Rightarrow F(v_{(\alpha-1)(\omega+2)}) =$ rot.

Wegen $v_0, v_{(\alpha-1)\omega+2} \in E(G')$ ist F keine gute Färbung. Dieser Widerspruch zeigt, daß die chromatische Zahl von G' zumindest ω ist, also ist der Subgraph G' von G nicht perfekt. $\square$

Bemerkung. Nicht jeder (ω, α)-zerlegbare Graph ist ein $C^{\omega-1}_{\omega\alpha+1}$.

Beispiel.



17. Vorlesung, 16. Januar 96
Mitschrift: Carola Wenk

Lemma 5. *Sei $G = (V, E)$ (ω, α)-zerlegbar mit $|V| = n$ und sei B die im Beweis von Satz 6 definierte Matrix. Dann entspricht jeder α-unabhängigen Menge von G eine Zeile von B.*

Beweis. Es gilt $(\frac{1}{\alpha} \cdot J - A) \cdot B^\top = \frac{1}{\alpha}\alpha \cdot J - A \cdot B^\top = J - (J - I) = I$. Damit ist noch einmal gezeigt, daß B nicht-singulär ist. Genauer $(B^\top)^{-1} = (\frac{1}{\alpha} \cdot J - A)$.

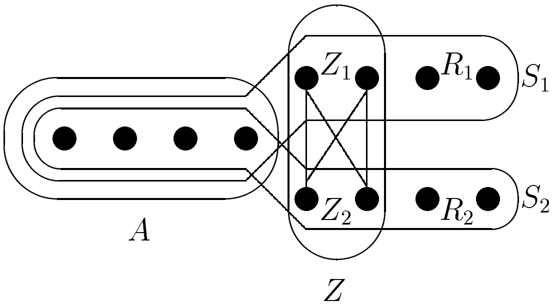
Sei nun S eine α-unabhängige Menge und $t = (B^\top)^{-1} \cdot \mathbb{1}_S$. Dann gilt $t = (\frac{1}{\alpha} \cdot J - A) \cdot \mathbb{1}_S = \mathbb{1} - A \cdot \mathbb{1}_S$. Da jede Zeile von A charakteristischer Vektor einer Clique C ist, und $\mathbb{1}_C^\top \cdot \mathbb{1}_S \in 0, 1$ ist, ist t ein 0,1-Vektor.

Nun zählen wir die Einsen in t: $\mathbb{1}^\top \cdot t = \mathbb{1}^\top(\mathbb{1} - A \cdot \mathbb{1}_S) = n - \mathbb{1}^\top \cdot A \cdot \mathbb{1}_S = n - \omega \cdot \mathbb{1}^\top \cdot \mathbb{1}_S = n - \omega\alpha = (\omega\alpha + 1) - \omega\alpha = 1$. Aus $B^\top \cdot t = \mathbb{1}_S$ folgt nun, daß $\mathbb{1}_S$ eine Zeile von B ist. $\square$

Bemerkung. Analog zeigt man, daß unter obigen Voraussetzungen jede ω-Clique eine Zeile in A repräsentiert.

Lemma 6. *Sei G ein (ω, α)-zerlegbarer Graph und S_1 und S_2 unabhängige Mengen der Größe α in G. Dann ist $G[S_1 \triangle S_2]$ zusammenhängend (mit $\triangle$ bezeichnen wir die symmetrische Differenz zweier Mengen).*

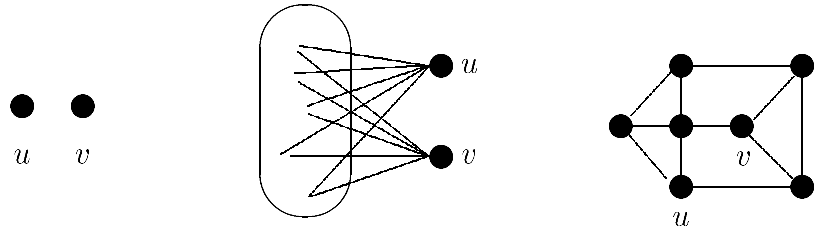
Beweis. Indirekt. Seien S_1 und S_2 zwei α-unabhängige Mengen, und angenommen, $G' := G[S_1 \triangle S_2]$ ist unzusammenhängend. Sei Z eine Zusammenhangskomponente von G'. Definiere $A := S_1 \cap S_2$ und $Z_i := Z \cap S_i$ und $R_i := S_i \setminus (A \cup Z_i)$ für $i = 1, 2$. Dann sind die beiden Mengen $T_1 := A \cup Z_1 \cup R_2$ und $T_2 := A \cup Z_2 \cup R_1$ unabhängig, und nach Konstruktion gilt $|T_1| + |T_2| = 2\alpha$, weshalb T_1 und T_2 beides α-unabhängige Mengen sind. Nun sind jedoch nach obigem Lemma $\mathbb{1}_{S_1}^\top, \mathbb{1}_{S_2}^\top, \mathbb{1}_{T_1}^\top$ und $\mathbb{1}_{T_2}^\top$ Zeilen von B, aber es gilt $\mathbb{1}_{T_2} = \mathbb{1}_{S_1} + \mathbb{1}_{S_2} - \mathbb{1}_{T_1}$, was einen Widerspruch zur Nicht-Singularität von B darstellt.



□

Definition. Zwei Knoten u,v in einem Graphen G heißen *even pair*, wenn es keinen induzierten (= sehnenfreien) Pfad mit ungerade vielen Kanten zwischen u und v gibt.

Beispiele. 1) 2) 3)



Lemma 7. (ω, α)*-zerlegbare Graphen besitzen keine even pairs.*

Beweis. Seien $x, y \in V$ beliebig. Aus der (ω, α)- Zerlegbarkeit des Graphen folgt, daß es zwei α -unabhängige Mengen S_x und S_y mit $x \in S_x$, $y \notin S_x$, $x \notin S_y$ und $y \in S_y$ gibt. Nach obigem Lemma ist der Graph $G' := G[S_x \triangle S_y]$ zusammenhängend und bipartit. Ein kürzester Weg zwischen x und y ist ein ungerader sehnenfreier Weg in G', also auch in G. $\Rightarrow x, y$ ist kein *even pair.* □

Satz 8. *Ein Mip besitzt kein even pair.*

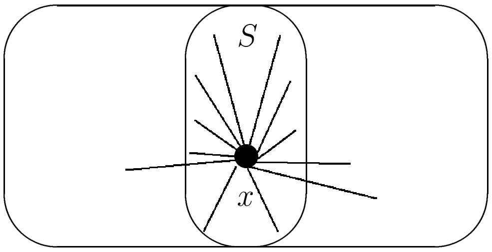
Beweis. Da Mips (ω, α)-zerlegbar sind, folgt der Satz aus Lemma 10. □

Definition. Eine Menge $S \subset V$ heißt *trennender Stern*, wenn

(1) $G[V - S]$ ist nicht zusammenhängend.

(2) es existiert ein $x \in S$, *Zentrum* genannt, so daß $S \subseteq N[x]$. ($N[x]$ ist die abgeschlossene Nachbarschaft von x).

Beispiel.



Satz 9. *Ein Mip besitzt keinen trennenden Stern.*

Beweis. Indirekt. Sei G ein Mip mit trennendem Stern S und Zentrum $x \in S$. Sei W eine Zusammenhangskomponente von $G[V - S]$ und $G_1 := G[W + S]$, $G_2 := G[V - W]$. Dann sind G_1 und G_2 perfekt. Seien nun F_i Färbungen der G_i mit $\omega(G_i)$ Farben. Sei I_i die Menge der Knoten in G_i, die dieselbe Farbe wie x haben und definiere $I := I_1 \cup I_2$. Da S trennend ist, existiert keine Kante zwischen I_1 und I_2. Als Farbklassen sind I_1 und I_2 unabhängig. Weil x Zentrum besteht $I \cap S$ nur aus x. Damit ist gezeigt, daß I unabhängig in G ist.

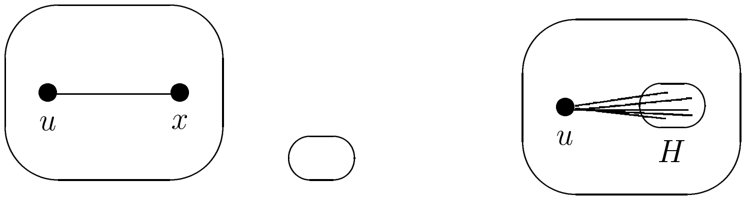
Da S trennend ist, ist jede ω-Clique C von G ganz in G_1 oder G_2 enthalten. Weil sowohl G_1 als auch G_2 perfekt sind ist $C \cap I \neq \emptyset$, also $\omega(G[V - I]) < \omega(G)$. Das bedeutet, daß $G[V - I]$ nicht perfekt ist. Ein Widerspruch zur Annahme G Mip. □

8.3 Operationen, die Perfektheit erhalten

Als Beispiel einer Operation, die Perfektheit erhält kennen wir die Multiplikation von Knoten: Ist G perfekt und w eine Abbildung $w : V \to \mathbb{N}$, dann ist $G \circ w$ ebenfalls perfekt.

Definition. G, H Graphen und $x \in V(G)$. Dann ist G_x^H der Graph mit der Knotenmenge $(V(G) - x) \cup V(H)$ und der Kantenmenge $E(G - x) \cup E(H) \cup \{(u, v) : u \in N_G(x), v \in V(H)\}$. Man sagt G_x^H ist durch *Substitution* von x durch H in G entstanden.

Beispiel.



G H G_x^H

Lemma 10. *Substitution erhält Perfekheit (d.h. G, H perfekt $\Rightarrow G_x^H$ perfekt)*

Beweis. Indirekt. Angenommen, G_x^H ist nicht perfekt. Dann existiert ein Mip M, der ein induzierter Subgraph von G_x^H ist. Es existieren induzierte Subgraphen G' von G und H' von H, so daß $M = G'^{H'}_x$, (die Graphen G' und H' sind durch die Knotenmengen $V(G') = V(M) \cap V(G) + x$ und $V(H') = V(M) \cap V(H)$ induziert).

Betrachte folgende Eigenschaften

(a) $\exists y \in G' : (yx) \notin E(G)$

(b) $\exists u, v \in H' : (uv) \notin E(H)$

Angenommen, (a) ist nicht erfüllt. Dann ist $\overline{M}$ unzusammenhängend, was einen Widerspruch zu den Eigenschaften eines Mips darstellt. Also gilt (a).

Angenommen, (b) ist nicht erfüllt. Dann ist H' eine Clique in G'. Wenn $|V(H')| \geq 2$, dann existieren Cozwillinge in M. Widerspruch. Wenn $|V(H')| \leq 2$, dann ist M isomorph zu einem Subgraphen von G. Widerspruch.

Wenn (a) und (b) erfüllt sind, dann ist $N[u]$ ein trennender Stern. Im Widerspruch dazu, daß M ein Mip ist. $\square$

18. Vorlesung, 23.Januar 96
Mitschrift: Christof Schultz

Lemma 11. *Wenn G P_4-frei und $|V| > 1$ ist folgt:*

(1) *G enthält Zwillinge oder*

(2) *G enthält Cozwillinge.*

Beweis. Der Fall $|V| = 1$ ist klar. Betrachten wir nun $|V| > 1$: Wenn G eine Clique ist, ist jedes Knotenpaar ein Cozwillingspaar. Seien nun x, y Knoten aus G mit $(x, y) \notin E$ und A eine minimal trennende Menge für x und y. Dann zerfällt $G[V - A]$ in mehrere Komponenten, wobei C_x die Komponente ist, die x enthält und C_y die die y enthält.

Behauptung: Für alle $z \in C_x$ und alle $a \in A$ ist $(z, a) \in E$. Angenommen das gilt nicht. Sei $K = (z \ldots x' \ldots a \ldots y)$ der kürzesten Weg von z über a nach y. Das ist ein sehnenfreier Weg mit mehr als 4 Knoten. Dies steht aber im Widerspruch dazu, daß der Graph P_4-frei sein soll.

Also ist jeder Knoten aus $C_x \cup C_y$ mit jedem Knoten aus A verbunden. Betrachten wir nun zwei Fälle:

1. Wenn $|C_x| = |C_y| = 1$ ist, folgt unmittelbar, daß x und y Zwillinge sind.
2. Wenn $|C_x| \geq 2$ folgt, mit obiger Überlegung, daß ein Zwillings-, bzw. ein Cozwillingspaar in $G[C_x]$ ein ebensolches Paar in G is. □

Folgerung. P_4-freie Graphen sind perfekt.

Beweis. P_4-Freiheit ist eine heritäre Eigenschaft, d.h. ein nicht perfekter P_4-freier Graph besitzt eine P_4-freien Mip. Nach obigen Lemma besitzt dieser Mip aber Zwillinge, bzw. Cozwillinge, was aber im Widerspruch zu den Eigenschaften eines Mip steht. □

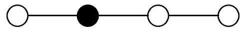
Übung 6. *Wenn G P_4-frei ist, dann ist G oder $\overline{G}$ unzusammenhängend.*

Übung 7. *P_4-freie Graphen sind Vergleichbarkeitsgraphen.*

8.4 Rot–Weiß Dekompositionssätze

Im folgendem betrachten wir Graphen mit einer Rot/Weiß-Färbung, die nicht unbedingt eine gute Färbung sein muß. Speziell interessiert uns die auf P_4's in G induzierten Färbungen.

Beispiel.

 P_4 vom Typ $\texttt{WRWW}$, (es gilt $\texttt{WRWW} = \texttt{WWRW}$)

Allgemeine Form des Rot–Weiß Dekompositionssatzes
Wenn G so gefärbt ist, daß keine P_4 vom Typ $t \in T$ auftreten folgt: G ist genau dann perfekt, wenn beide einfarbigen Subgraphen perfekt sind. Hierbei ist T eine Menge von Färbungstypen.

Beispiel. Wenn jeder P_4 eine gerade Anzahl von roten Knoten hat folgt: G ist genau dann perfekt, wenn $G_{\text{weiß}}$ und G_{rot} perfekt sind. Hierbei ist $T = \{\texttt{WRRR}, \texttt{RWRR}, \texttt{RWWW}, \texttt{WRWW}\}$ (Beweis in [8]).

Satz 12. *Für die folgenden Mengen T_1, T_2, T_3 gilt der Dekompositionssatz:*

(1) $T_1 = \{\texttt{RRRR}, \texttt{RWWR}, \texttt{WRRW}, \texttt{WWWW}\}$

(2) $T_2 = \{\texttt{WRRR}, \texttt{RWRR}, \texttt{RWWR}, \texttt{WRRW}, \texttt{WWWW}\}$

(3) $T_3 = \{\texttt{WRRR}, \texttt{RWRW}, \texttt{RWWR}, \texttt{WRRW}, \texttt{WWWW}\}$

Beweis für (1) und (2) Angenommen es existiert ein Gegenbeispiel, dann auch ein minimales Gegenbeispiel, also ein Mip. Nach Vorraussetzung gibt es keinen weißen P_4 in G, somit auch nicht in $\overline{G}$. Aus dem Lemma 14 folgt, daß $G_{\text{weiß}}$ Zwillinge, bzw. Cozwillinge enthält. O.B.d.A nehmen wir an, daß $G_{\text{weiß}}$ die Zwillinge u, v enthält (ansonsten arbeite mit $\overline{G}$ weiter).

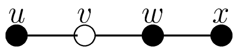
Behauptung: u, v ist ein even pair in G, d.h. jeder von u, v induzierte Weg hat eine gerade Anzahl von Kanten. Sei $P = (u = w_0 w_1 w_2 \ldots w_k = v)$ ein sehenfreier Weg zwischen u und v. $k = 1$ geht nicht, da $uv \notin E$, für $k = 2$ ist die Behauptung erfüllt. Betrachten wir nun $k \geq 2$. Hier gilt, da u und v Zwillinge in $G_{\text{weiß}}$ sind, daß w_1 und $w_k - 1$ rot sind. Betrachten wir nun die max. monochromatischen Teilstücek von P:

Für T_1: Da T_1 symmetrisch bezüglich der Vertauschung von $\texttt{R}$ und $\texttt{W}$ in den Färbungstypen ist, genügt es die weißen Teilstücke $\texttt{RW}^i\texttt{R}$ zu betrachten. Da $\texttt{WWWW} \in T_1$ gibt es keine weißen Teilstücke mit $i \geq 4$. Teilstücke der Länge 2 fallen auch weg, da $\texttt{RWWR} \in T_1$. Also haben alle max. monochromatischen Teilstück eine ungerade Länge, nämlich 1 oder 3.

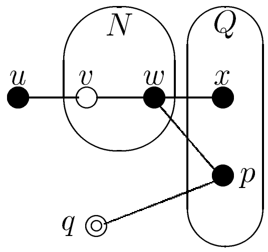
Für T_2: Die Argumentation für die weißen Teilstücke gehen analog zu obigen. Betrachten wir nun roten Teilstücke. Rote Teilstücke der Länge 2 sind verboten, da $\texttt{WRRW} \in T_2$ ist. Teilstück mit eine Länge größer als 3 sind auch verboten, da das Anfangsstück WRRR $\in T_2$ ist. Also gibt es nur rote Teilstücke der Länge 1, d.h. mit ungerader Länge.

Das bedeutet, daß zwischen u und v eine ungerade Anzahl von Knoten liegen, d.h. P enthält eine gerade Anzahl von Kanten. Somit wäre aber u, v ein even pair, was im Widerspruch dazu steht, daß G ein Mip sein sollte. $\square$

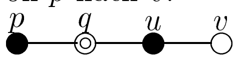
Beweis für (3) Wie im obigen Beweis sei G ein minimales Gegenbeispiel, also Mip. Wenn kein P_4 in G vom Typ RWRR ist, dann folgt die Aussage aus Teil (2), da $T_2 \subseteq T_3 +$ RWRR. Betrachten wir nun im Folgenden einen $P_4 : (u, v, w, x)$ vom Typ RWRR:



Sei nun $N := N(v) + v - u$ und Q die Menge der rote Nachbarn von w, die nicht zu v und nicht zu u benachbart sind ($x \in Q$). Angenommen es existiert ein Paar p, q mit $p \in Q$ und $q \notin N \cup Q$ und $(q, p) \in E$.

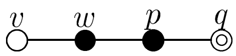


1.Fall: Angenommen $(q, u) \in E$. Wegen $p \in Q$ ist $(p, u) \notin E$, wegen $p, q \notin N$ sind $(q, v) \notin E$ und $(p, v) \notin E$. Also ist (p, q, u, v) ein sehnenfreier Pfad von p nach v.



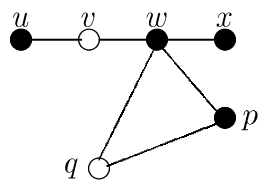
Dies geht nicht, da RXRW für X $\in \{$R, W$\}$verboten ist, d.h. $(q, u) \notin E$.

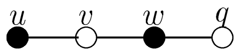
2.Fall: Betrachte nun den Pfad (v, w, p, q). Nach Voraussetzung is $(v, p) \notin E$ und $(v, q) \notin E$. Wenn $(q, w) \notin E$, dann ist (v, w, p, q) ein sehnenfreier Pfad von v nach w.



Dies ist verboten, da WRRX für X $\in \{$R, W$\}$ verboten ist. Also ist $(q, w) \in E$.

Nachdem die Kante $(q, w) \in E$ und $q \notin N \cup Q$, folgt, daß q weiß ist.



Betrachten wir nun: 

Das ist ein induzierter P_4. Da `RWRW` verboten ist, kann es keine solchen Knoten p, q geben. Deshalb zerfällt $G[V - N]$ in mehrere nicht leere Komponenten C_u und C_x. Das heißt aber, daß N ein trennender Stern mit Zentrum v ist, was im Widerspruch dazu steht, daß G ein Mip sein sollte. $\square$

Zum Kapitel

Der Beweis von Satz 4 stammt aus Lovász *a characterization of perfect graphs* und ist in [2] (Seiten 40–42) nachgedruckt. Dort findet man auch den ursprünglichen Beweis des PGT der die Theorie der Hypergraphen verwendet. Fulkerson hatte zuvor schon eine Theorie der Antiblocker von Polyeder in der zum Beweis des PGT nur der Beweis fehlte, daß Substitution Perfektheit erhält.

Aufbauend auf der für Mips gültigen Gleichung $n = \omega\alpha + 1$ wurde diese Klasse von Padberg sowie von Bland, Huang und Trotter genauer Untersucht. Originalarbeiten und weitere Ergebnisse zu Mip und (ω, α)-zerlegbaren Graphen findet man in [2].

Der Begriff des even-pairs stammt von Meyniel, der hier vorgestellte Beweis von Lemma 10 ist von Reed [41]. Meyniel [35] definiert *quasi parity* Graphen als jene Graphen G in denen für jedes $W \subseteq V$ entweder $G[W]$ oder $\overline{G[W]}$ ein even-pair besitzt. Quasi parity Graphen sind eine große Klasse perfekter Graphen, sie umfassen Meyniel-Graphen und $\mathcal{PO}$ Graphen. Die grundlegende und sehr lesenswerte Arbeit über trennende Sterne stammt von Chvátal [7]. Dort werden auch weitere Operationen die Perfektheit erhalten diskutiert.

P_4-frei Graphen heißen auch *Cographen*, sie sind die Vergleichbarkeitsgraphen von *Serien-Parallelen Ordnungen* und treten als besonders gut verstandene Klasse von Graphen in den verschiedensten Zusammenhängen auf (vergl. [5]). Rot-Weiß Dekompositionssätze wurden zuerst von Chvátal und Hoàng [8] und [25] untersucht. Die hier behandelten Fälle und die vollständige Charakterisierung der Mengen von Färbungstypen für die der Rot-Weiß Dekompositionssatz gilt ist von Chvátal, Lenhart und Sbihi [9].

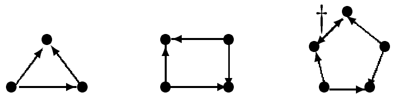
9 Vergleichbarkeitsgraphen

Vergleichbarkeitsgraphen sind als die zugrundeliegenden Graphen von Ordnungen definiert, das heißt, ein ungerichteter Graph $G = (V, E)$ ist ein Vergleichbarkeitsgraph, wenn ein orientierter Graph $H = (V, F)$ mit $E = \{\{u, v\} : (uv) \in F$ oder $(vu) \in F\}$ und F transitiv existiert. (H orientierter Graph bedeutet $F \cap F^{-1} = \emptyset$ und F transitiv bedeutet $F^2 \subseteq F$, bzw. $(x, y) \in F$ und $(y, z) \in F \Rightarrow (x, z) \in F$). Erfüllt F die genannten Bedingungen, so heißt F *transitive Orientierung* von G.

Bemerkung. Wenn F eine transitive Orientierung von G ist, so auch F^{-1}.

Folgerung. Für eine Kante aus E dürfen wir die Orientierung beliebig vorgeben.

Beispiel.



† kein Vergleichbarkeitsgraph

9.1 Die Γ Relation

19. Vorlesung, 25.01.96
Mitschrift: Bernd Falkenberg

Definition. Die Γ-*Relation (forcing relation)* ist eine Relation auf der Menge $\hat{E}$ der geordneten Paare uv für $\{u, v\} \in E$:

$$uv \; \Gamma \; u'v' \Leftrightarrow \begin{cases} u = u' \text{ und } \{v, v'\} \notin E \text{ oder} \\ v = v' \text{ und } \{u, u'\} \notin E \end{cases}$$

Wir definieren die Relation Γ^* durch:

$$ab\Gamma^* cd \Leftrightarrow \exists x_i, y_i \text{ so daß } ab\Gamma x_1 y_1 \Gamma x_2 y_2 \Gamma .. \Gamma x_k y_k \Gamma cd.$$ und

nennen so etwas eine Γ-*Weg* von ab nach cd.

Lemma 1. Γ^* *ist reflexiv, transitiv und symmetrisch.*

Beweis. Reflexiv: da Γ reflexiv ist.

Transitiv: Wenn es einen Γ-Weg von uv nach wx und einen von wx nach xy gibt, dann gibt es auch einen von uv nach xy, da man einfach die Wege aneinander hängen kann.

Symmetrisch Wenn es einen Γ-Weg von uv nach xy gibt. so kann man ihn einfach rückwärts laufen und erhält so einen Γ-Weg von xy nach uv. □

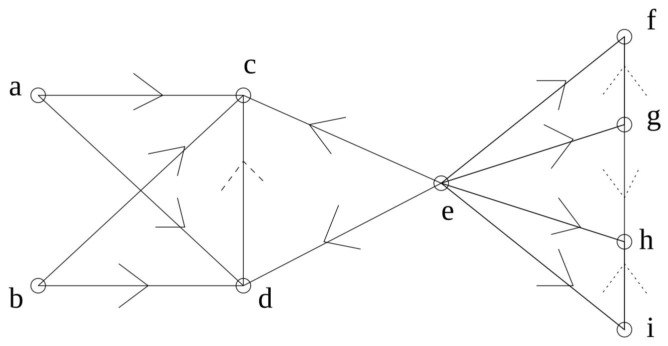
Also ist Γ^* eine Äquivalenzrelation und induziert eine Partition von $\hat{E}$. Die Klassen dieser Partition heißen *Implikationsklassen.*

Lemma 2. *I sei Implikationsklasse. Dann ist auch I^{-1} Implikationsklasse.*

Beweis. Seien ab, $cd \in I$. $ab\Gamma cd$. Dann gilt ba, $dc \in I^{-1}$. Wenn ab und cd den Endpunkt gemeinsam haben, so haben ba und dc den Anfangspunkt gemeinsam. Analog: ab und cd haben den Endpunkt gemeinsam.

Damit gilt : $ba\Gamma dc$ und somit folgt: I^{-1} ist Implikationsklasse. □

Beispiel.



In diesem Beispiel gibt es sechs Implikationsklassen. Diese sind : $I_1 = \{ac, bc, bd, ad, ec, ed, ef, eg, eh, ei\}$, $I_2 = \{dc\}$ und $I_3 = \{gf, gh, ih\}$. Die inversen Implikationsklassen I_1^{-1}, I_2^{-1} und I_3^{-1} sind die übrigen drei Implikationsklassen.

Lemma 3. *$I \subseteq \hat{E}$ Implikationsklasse, dann gilt $I = I^{-1}$ oder $I \cap I^{-1} = \emptyset$.*

Beweis. Aus $I \cap I^{-1} \neq \emptyset$ folgt $I = I^{-1}$, da die Implikationsklassen eine Partition darstellen. □

Satz 4. *$G = (V, E)$ hat transitive Orientierung F, dann gilt für jede Implikationsklasse $I \cap I^{-1} = \emptyset$.*

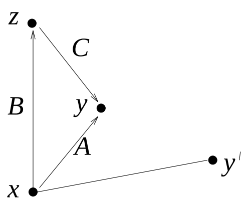
Beweis. : Sei $ab \in F \Rightarrow I_{(a,b)} \subseteq F$ und $ba \notin F \Rightarrow ba \notin I_{(a,b)} \Rightarrow ba \in I^{-1}_{(a,b)}$ (wegenLemma). □

Bemerkung. Wenn für jede Implikationsklasse I gilt $I \cap I^{-1} = \emptyset$, dann hat $G = (V, E)$ eine transitive Orientierung F. **Aber!** In einem vollständigen Graphen besteht jede Implikationsklasse aus nur einer Kante, also gilt nicht für jede Menge von Implikationsklassen $\{I_1, ..., I_k\}$ deren Vereinigung eine Orientierung von G ist, daß $\bigcup I_i$ transitiv ist.

Es folgen zwei technische Lemmata, die sogenannten Dreickslemmata.

Lemma 5. *Seien A, B, C Implikationsklassen mit $A \neq B, A \neq C$. Seien x, y, z Knoten mit $xy \in A, xz \in B, zy \in C$. Dann gilt:*

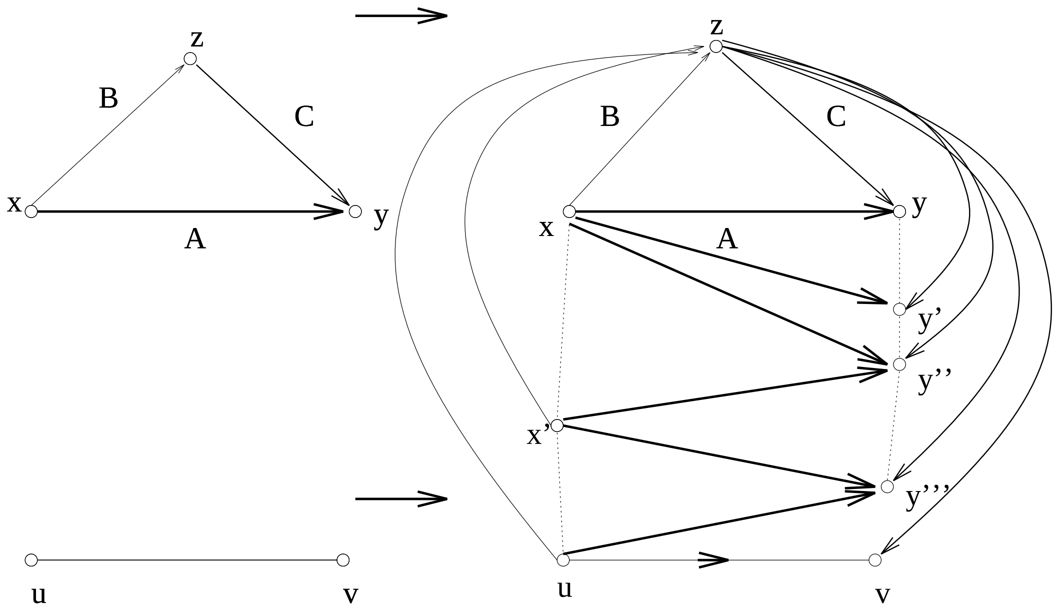
$$\forall y' \text{ mit } \{x, y'\} \in E, \{y, y'\} \notin E \text{ ist } zy` \in C. \text{ Insbesondere } \{xy'\} \in E.$$



Beweis. Zunächst ist klar, wenn $\{z, y'\} \notin E$ dann folgt aus $xy\Gamma xy'\Gamma xz$ schon $A = B$, Widerspruch. Aus $\{y, y'\} \notin E$ folgt $zy\Gamma zy'$, also wegen $zy \in C$ auch $zy' \in C$ □

Lemma 6. *Seien A, B, C Implikationsklassen mit $A \neq B, A \neq C$. Seien x, y, z Knoten mit $xy \in A, xz \in B, zy \in C$. Dann gilt: $uv \in A \Rightarrow uz \in B$ und $zv \in C$.*

Beweis. $xy \in A$ und $uv \in A$ bedeutet $xy\Gamma^* uv$, das heißt es gibt eine kürzeste Folge $xy\Gamma x_1 y_1 \Gamma ... x_k y_k \Gamma uv$. Wenn $k = 0$, also $xy\Gamma uv$ dann folgt die Behauptung aus dem vorigen Lemma. Als Induktionsvorausetzung dürfen wir nun annehmen, daß $x_k z \in B$ und $zy_k \in C$. Das Dreieck x_k, y_k, z hat nun dieselben Eigenschaften wie x, y, z und die Behauptung folgt wie eben aus dem vorigen Lemma. □



Wir verlassen das Thema transitive Orientierung nun zu einem Exkurs über Substitutionsdekomposition. Anschließend werden wir den engen Zusammenhang zwischen diesen beiden Konzepten kennenlernen.

9.2 Substitutionsdekomposition

Definition. Sei $W \subset V$, d.h. $W \neq V$, und $U = V - W$. Wenn für alle $w_1, w_2 \in W$ gilt: $N(w_1) \cap U = N(w_2) \cap U$ dann heißt W ein *Modul* von G. Wenn $|W| = 1$, dann heißt W *trivial*. Sonst heißt W *nichttriveal*.

Im Abschnitt 8.3 hatten wir Substitution als Operation kennengelernt, die einen Knoten eines Graphen G durch einen anderen Graphen H ersetzt, und jeden Knoten aus H mit den Nachbarn von x verbindet, der so erhaltene Graph wurde mit G_x^H bezeichnet. Offenbar ist $V(H)$ also ein Modul in G_x^H.

Die Umkehrung gilt auch: Sei W Modul in G, $w_0 \in W$ beliebig und $U = V(G) - V(W)$. Dann lässt sich G auch schreiben als : $G[U + w_0]_{w_0}^{G[W]}$.

Definition. Wenn G ein nichttriviales Modul besitzt oder $|V(G)| \leq 2$, dann heißt G *dekomponierbar*. Andernfalls heißt G *prim*.

Übung 1. *G prim $\Rightarrow$ G und $\overline{G}$ sind zusammenhängend.*

Satz 7. *(Dekompositionssatz) Für jeden Graphen G gilt genau eine der folgenden Aussagen:*

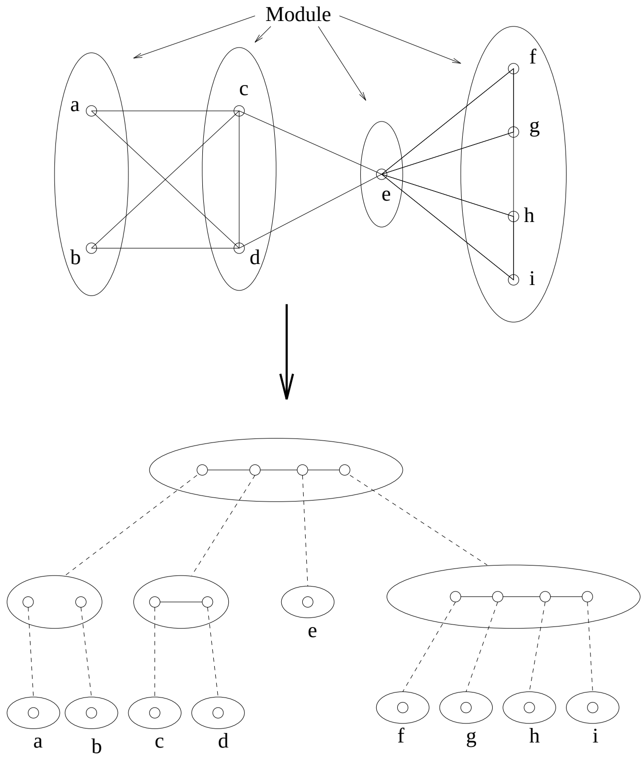
(1) *G ist unzusammenhängend (paralleler Fall)*

(2) *$\overline{G}$ ist unzusammenhängend (serieller Fall)*

(3) *$\exists!$ primer Graph H mit Knoten $v_1, ..., v_k \in V(H)$ und Graphen $K_1, ..., K_k$, sodaß $G = H\,{}^{K_1\ K_2 \dots\ K_k}_{v_1\ v_2 \dots\ v_k}$.*

Wir werden den Satz später beweisen. Zunachst stellen wir fest, daß basierend auf den drei Fällen aus dem Satz jedem Graphen G ein *Dekompositionsbaum* T zugeordnet werden kann. Jedem Knoten von T ist ein induzierter Subgraph von G und ein Typ aus {parallel,seriell,prim} zugeordnet, so daß der Typ gerade der auf den Subgraphen zutreffende Fall ist. Die Wurzel von T ist G. Die Nachfolger von paralellen und seriellen Knoten sind die Zusammenhangskomponenten beziehungsweise die Zusammenhangskomponenten des Komplements. Nachfolger von primen Knoten sind die Graphen $K_1, ..., K_k$.

Beispiel. Ein Graph und sein Dekompositionsbaum.



Übung 2. *Wie verhält sich der Dekompositionsbaum von $\overline{G}$ zum Dekompositionsbaum von G?*

Bevor nun der Dekompositionssatz bewiesen werden soll, zunächst einige Lemmata über Module:

Lemma 8. *Seien M_1, M_2 Module, $M_1 \cap M_2 \neq \emptyset$. Dann gilt: $M_1 \cup M_2 = V$ oder $M_1 \cup M_2$ ist ein Modul. Außerdem ist $M_1 \cap M_2$ ein Modul.*

Beweis. Sei $M_1 \cup M_2 \neq V$ und $m \in V \backslash (M_1 \cup M_2)$. Da M_1 Modul, verhalten sich je zwei Knoten aus M_1 zu m gleich. Ebenso mit M_2. Seien jetzt $m_1 \in M_1, m_2 \in M_2$. Da $M_1 \cap M_2 \neq \emptyset$ gilt, gibt es einen Knoten $m_3 \in M_1 \cap M_2$. Wenn dieser eine Kante zu m besitzt, so auch m_1 und m_2. Wenn keine Kante zwischen m und m_3 exitstiert, so auch keine zwischen m_1 und m sowie m_2 und m. Da m_1 und m_2 beliebig waren $\Rightarrow M_1 \cup M_2$ ist ein Modul.

Seien $m_1, m_2 \in M_1 \cap M_2$. Da $m_i \in M_1$, verhalten sie sich gegenüber allen Knoten in $V \backslash M_1$ gleich. Ebenso verhalten sie sich allen Knoten in $V \backslash M_2$ gleich. Also verhalten sie sich allen Knoten aus $V \backslash (M_1 \cap M_2)$ gegenüber gleich $\Rightarrow M_1 \cap M_2$ ist Modul. □

Lemma 9. *Seien M_1, M_2 Module, die nicht ineinander enthalten sind. Dann gilt: In G sind entweder alle oder keine Kanten zwischen $M_1 \backslash M_2$ und $M_2 \backslash M_1$ vorhanden.*

Beweis. Annahme, es gebe eine Kante $a_1 a_2$ mit $a_1 \in M_1 \backslash M_2$ und $a_2 \in M_2 \backslash M_1$. Da M_1 Modul ist folgt, daß jeder Knoten aus $M_1 \backslash M_2$ zu a_2 benachbart ist. Da M_2 Modul ist muß nun aber jeder Knoten aus $M_2 \backslash M_1$ genau wie a_2 zu jedem Knoten aus $M_1 \backslash M_2$ benachbart sein, das heißt, daß alle Kanten zwischen $M_1 \backslash M_2$ und $M_2 \backslash M_1$ existieren. □

Lemma 10. *Weder G noch $\overline{G}$ sind unzusammenhängend, M_1, M_2 sind maximale Module $\Rightarrow M_1 \cap M_2 = \emptyset$.*

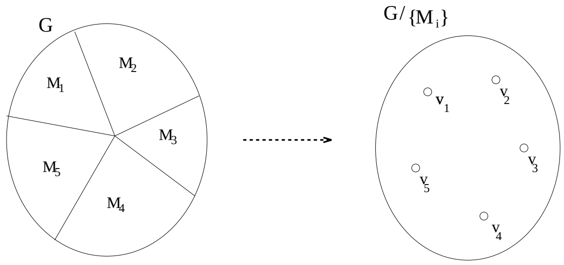
Beweis. Da M_1, M_2 maximale Module sind ist $M_1 \cup M_2$ kein maximales Modul, also wegen Lemma 8 $M_1 \cup M_2 = V$. Nun gilt mit Lemma 9: Entweder alle oder keine Kanten existieren zwischen $M_1 \backslash M_2$ und $M_2 \backslash M_1$. O.B.d.A. sei keine Kante zwischen $M_1 \backslash M_2$ und $M_2 \backslash M_1$ vorhanden (andernfalls gehen wir zum Komplementärgraphen über). Da M_1 Modul ist gibt es keine Kante zwischen M_1 und $M_2 \backslash M_1$. Wegen $M_1 \cup M_2 = V$ bedeutet das, daß G unzusammenhängend ist — Widerspruch. □

Beweis. (des Dekompositionssatzes) Annahme: G und $\overline{G}$ sind zusammenhängend. Dann gilt: Die maximalen Module $M_1, ..., M_k$ bilden eine Partition von V. Sei $K_i = G[M_i]$ und wähle $v_i \in K_i$ beliebig. Wir definieren den Quotientengraphen $G/_{\{M_i\}} := G[v_1, ..., v_k]$.
Behauptung: $G/_{\{M_i\}}$ ist das gesuchte H.

$G/_{\{M_i\}}$ ist zusammenhängend, da G zusammenhängend ist.

$\overline{G/_{\{M_i\}}}$ ist zusammenhängend, da $\overline{G}$ zusammenhängend ist.

Angenommen, es existiert ein nicht-triviales Modul. Dann gibt es eine Teilmenge der Module M_i, deren Vereinigung ein Modul in G ist. Das ist aber ein Widerspruch zur Maximalität der Module $M_1, ..., M_k$. □



Algorithmen zur Berechnung des Dekompositionsbaumes

- zuerst 70er Jahre $O(n^5), O(n^4)$
- Anfang der 80er $O(n^3)$ (Gallai, Golumbic, Buer & Möhring)
- Ende der 80er $O(n^2)$ (Muller & Spinrad)
- 1992 $O(m \log n)$ (Cournier & Habib)
- 1993 $O(m\, \alpha(m\, n))$ (Spinrad)
- 1994 $O(m + n)$ (McConnell & Spinrad)

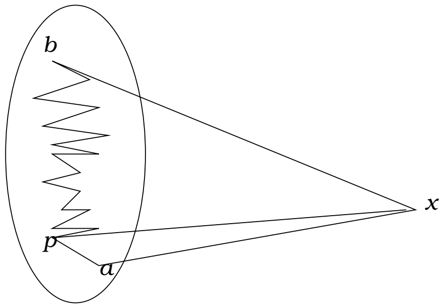
9.3 Transitive Orientierbarkeit und Substitution

20. Vorlesung, 30.1.96
Mitschrift: Susanna Gebauer

Wenn $I \subseteq \hat{E}$ eine Implikationsklasse ist, dann heißt $\hat{I} = I \cup I^{-1}$ *Farbklasse.* Farbklassen können als Teilmengen von E aufgefaßt werden. Der *Träger* einer Implikationsklasse I ist $Tr(I) = \{x : \exists y, \text{ so daß } xy \in \hat{I}\}$

Lemma 11. *Der Träger einer Implikationsklasse ist ein Modul.*

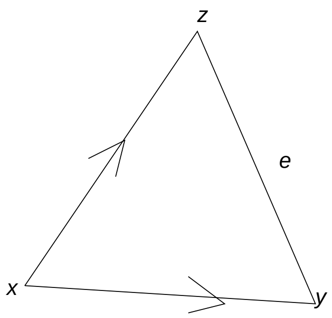
Beweis. Sei A Implikationsklasse und $\hat{A}$ die zugehörige Farbklasse. Seien $a, b \in Tr(A)$ und $x \notin TR(\hat{A})$. Wir müssen zeigen: $\{a, x\} \in E \iff \{b, x\} \in E$. Aus $a, b \in Tr(A)$ folgt es gibt einen Γ-Weg zwischen einer Kante die a enthält und einer die b enthält (siehe Bild).



Aus $\{ax\} \in E$ folgt $\{px\} \in E$, sonst wäre $ap\ \Gamma\ ax$ also $x \in Tr(A)$. Dieses Argument schreibt man fort bis schließlich $\{bx\} \in E$ $\square$

Lemma 12. *Keine zwei verschiedenen Farbklassen haben denselben Träger*

Beweis.



Sei $x \in Tr(\hat{A}) = Tr(\hat{B})$, d.h. x ist inzident mit zumindest einer Kante aus A und einer Kante aus B, also z.B. $ay \in \hat{A}$ und $xz \in \hat{B}$. Die Kante $e = \{yz\} \in E$, weil $A \neq B$.

Fall 1. $e \notin A$, dann folgt mit Lemma 5: keine Kante aus A ist mit z inzident (andernfalls müßte $\{z, z\}$ in $\hat{B}$ sein). Also $Tr(A) \neq Tr(B)$ – Widerspruch.

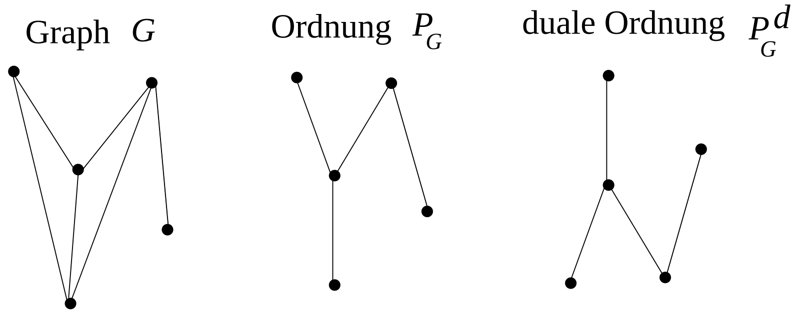
Fall 2. $e \in \hat{A}$ also $e \notin \hat{B}$, dann folgt symmetrisch zu Fall 1 der Widerspruch $Tr(A) \neq Tr(B)$ $\square$

Satz 13. *G prim $\Rightarrow$ in G gibt es nur eine Farbklasse.*

Beweis. Wenn G prim, dann gibt es keine nichttrivialen Module. Aus Lemma 11 folgt daher, daß jede Farbklasse Träger V hat. Nun folgt mit Lemma 12, daß es nur eine Farbklasse gibt. □

Folgerung. G primer Vergleichbarkeitsgraph $\Rightarrow$ es gibt genau ein Paar F, F^{-1} von transitiven Orientierungen für G. Die Orientierung F^{-1} heißt die zu F *duale Orientierung.*

Beispiel.



Der Satz zeigt, daß es einfach ist prime Vergleichbarkeitsgraphen, transitiv zu orientieren (starte mit beliebiger Kante und bilde die Implikationsklassen). Wenn der Dekompositionsbaum des Graphen gegeben ist, dann ist es auch für nicht-prime Vergleichbarkeitsgraphen einfach eine transitive Orientierung zu finden.

Bemerkung.

1. Fast alle Graphen sind prim.
2. Fast alle Vergleichbarkeitsgraphen sind prim.

Übung 3. *Trivialerweise findet man zu einer Kante uv alle xy mit $uv \; \Gamma \; xy$ mit $O(n^2)$ Operationen. Zeige, daß schon $O(n)$ Operationen genügen. Verwende dieses Teilergebnis im Entwurf eines $O(mn)$ Algorithmus zur transitiven Orientierung primer Vergleichbarkeitsgraphen.*

9.4 Schnelle transitive Orientierung

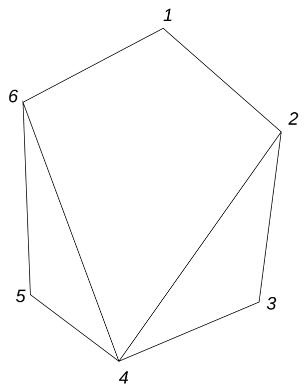
In diesem Abschnitt wollen wir einen $O(m \log n)$ Algorithmus zur transitiven Orientierung primer Vergleichbarkeitsgraphen kennenlernen. Die Beschränkung auf prime Graphen ist durch die am Ende des vorigen Abschnitts gemachten Bemerkungen gerechtfertigt.

Sei G ein primer Vergleichbarkeitsgraph im ersten Schritt suchen wir ein Element $c \in V$ das in einer transitiven Orientierung von G ein Maximum ist. Der nachfolgende Algorithmus CandSplit findet so ein Element.

CandSplit(G, v)

  $C \leftarrow V - v$
  $Q \leftarrow v$
  repeat
    $x \leftarrow$ pop Q
    if $C \cap N(x) \neq \emptyset$ **then**
      $Q \leftarrow Q + [C - N(x)]$
      $C \leftarrow C \cap N(x)$
  until $\mid C \mid = 1$
  return C

Beispiel.



Die folgende Tabelle illustriert wie CandSplit(G, 6) auf dem Beispielgraphen arbeitet. Die erste Spalte zeigt das soeben bearbeitete *Splitelement* x, die zweite die Kandidatenmenge C, die dritte die Queue Q.

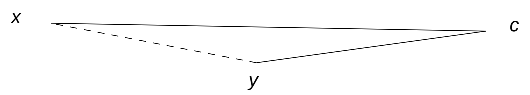
	$\{1,2,3,4,5\}$	$[6]$
6	$\{1,4,5\}$	$[3,2]$
3	$\{1,4\}$	$[2,5]$
2	$\{4\}$	$[5,1]$

return 4

Lemma 14. *DerAlgorithmus erhält die Invariante:*

$\forall c \in C$ *und alle* $y \notin C$ *gilt: wenn* $\{c, y\} \in E$, *dann ist* $(y, c) \in I_{(v,c)}$.

Beweis. Ganz zu Beginn ist v das einzige mögliches y und die Aussage ist trivial. Sei nun $c, y \neq v$ ein Paar, das den Bedingungen genügt $\Rightarrow$ es gab ein Splitelement x, das y aus C herausholte.



Die Invaraiante impliziert,daß x und y nicht benachbart sind und $(x,c) \in I_{(v,c)}$. Aus $(y,c)\Gamma(x,c)$ folgt wie behauptet $(y,c) \in I_{(v,c)}$. □

Satz 15. CANDSPLIT *ist korrekt, d.h. wenn G primer Vergleichbarkeitsgraph ist und F eine transitive Orientierung von G, dann ist das berechnete Element entweder ein Maximum oder ein Minimum in F.*

Beweis. Da G prim gibt es genau ein Paar dualer Implikationsklassen die als $I_{(xy)}$ und $I_{(yx)}$ für beliebiges $\{xy\} \in E$ geschrieben werden können. Wenn CANDSPLIT ein c zurückgibt, dann impliziert das Lemma, daß c Maximum in $I_{(vc)}$ ist.

Bleibt zu überprüfen, daß der Algorithmus tatsächlich ein c zurückgibt. Es gilt:

- $|C| \geq 1$ das wird durch die Abfrage $C \cap N(x) \neq \emptyset$ sichergestellt.
- Nachdem $x \in V$ als Splitelement verwendet wurde ist x weder in C noch in Q. Damit ist ausgeschlossen, daß der Algorithmus in eine unendliche Schleife gerät.

Als Fehlermöglichkeit bleibt nun noch, daß der pop Befehl auf eine leere Queue angewendet wird. In diesem Fall ist $|C| > 1$ und für alle $x \notin C$ gilt x wurde als Splitelement verwendet, das heißt entweder ist $C \cap N(x) = \emptyset$ oder $C \subseteq N(x)$. Also ist C ein nichttriviales Modul und G ist als illegale Eingabe enttarnt. □

Lemma 16. CANDSPLIT *kann so implementiert werden, daß die Laufzeit $O(n+m)$, also linear ist.*

Beweis. Wenn wir annehmen, daß die Mengen $C \cap N(x)$ und $C - N(x)$ in Zeit proportional zu $|N(x)|$ berechnet werden können, dann ist die angegebene Laufzeitschranke leicht einzusehen.

Die Annahme kann realisiert werden, indem man C als doppelt verkettete Liste hält und dafür sorgt, daß jedes Element $y \in N(x)$ in konstanter Zeit in C lokalisiert und aus C gelöscht werden kann. □

21. Vorlesung, 6. Februar 96
Mitschrift: Johannes Schwarz

Nachdem wir für einen primen Vergleichbarkeitsgraphen G den Algorithmus CandSplit(G, v) entworfen haben, der uns einen Knoten c liefert, der Maximum in einer transitiven Orientierung F von G ist, wird der nächste Algorithmus die geordnete Partition $(V\backslash\{c\}, \{c\})$ verfeinern. Das Ziel ist die Partition so zu Verfeinern, daß die zuletzt erreichte Verfeinerung eine lineare Erweiterung der transitiven Orientierung F ist. Ist eine lineare Erweiterung L von F berechnet, dann ist Kante $\{x, y\}$ in F als (x, y) orientiert, genau wenn x vor y in L, also ist F durch L vollständig bestimmt.

Das angegebene Ziel wird erreicht, wenn die geordnete Partition $(V - c, c)$ schrittweise verfeinert wird, so daß gilt:

(1) **Invariante.** Kanten zwischen Blöcken sind von links nach rechts orientiert. (Etwas formaler: Seien B, B' Blöcke und B vor B' in der geordneten Partition, für alle $x \in B$ und $y \in B'$ mit $\{x, y\} \in E$ gilt $(x, y) \in F$).

(2) Die Partition kann so lange verfeinert werden, bis jeder Block einelementig ist.

Von der folgenden Operation werden wir zeigen, daß sie die angegebenen Eigenschaften erfüllt. Sei $s \in V$ beliebig, wir nennen S das *Splitelement.*

Split(s)

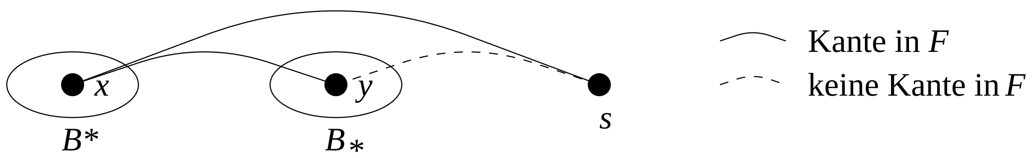
Für jeden Block B mit $s \notin B$ seien $B^* := B \cap N(s)$ und $B_* := B\backslash B^*$.

- Ist B vor dem Block von s in der Partition, dann ersetze B durch B^*,B_*.
- Ist der Block von s vor B in der Partition, dann ersetze B durch B_*,B^*.

Lemma 17. *Der Split–Operation erhält die Invariante.*

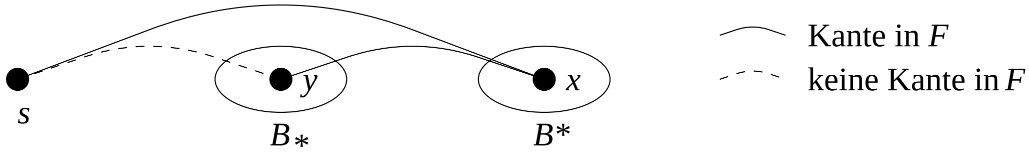
Beweis. Seien $x, y \in B$, $\{x, y\} \in E$ und x,y liegen nach dem Split mit s in verschiedenen Blöcken.

1. Fall: x liegt vor s, $x \in N(s)$, $y \notin N(s)$.



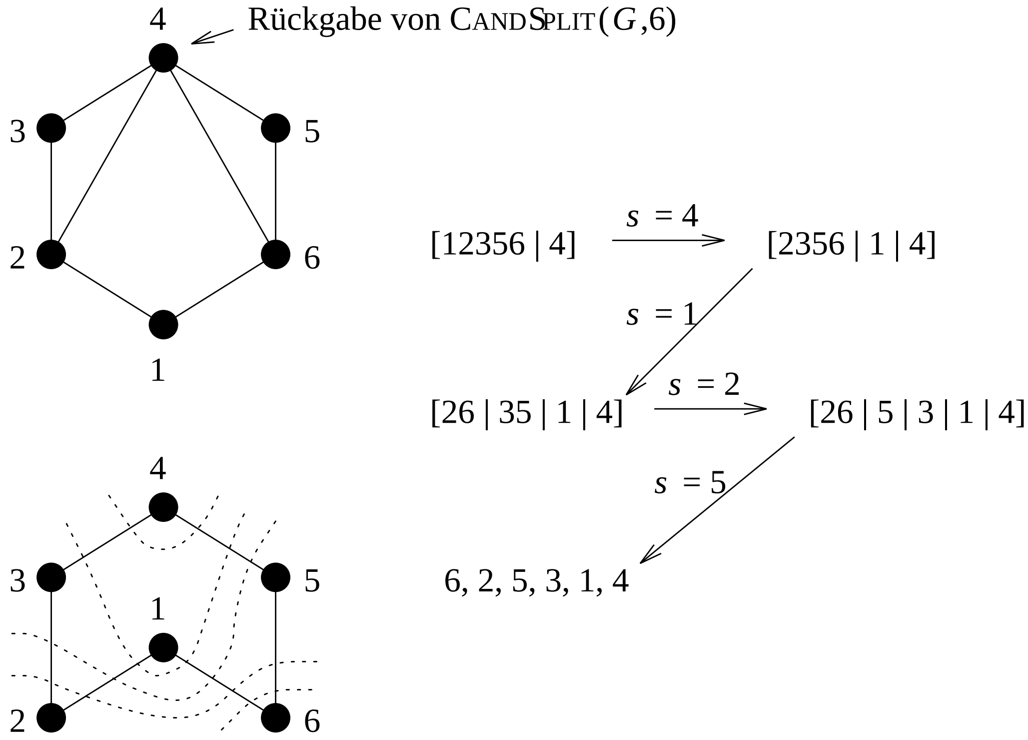
Also ist $xs\Gamma xy$, und damit folgt aus $xs \in F$ auch $xy \in F$.

2. Fall: x liegt hinter s, $x \in N(s)$, $y \notin N(s)$.



Also folgt analog zum 1. Fall $yx \in F$. □

Beispiel Wir betrachten eine Folge von Split–Operationen an dem Graphen an dem wir auch CandSplit veranschaulicht haben.



Als Ergebnis erhalten wir die lineare Erweiterung $L = (6, 2, 5, 3, 1, 4)$ aus der wir $F = \{65, 61, 64, 23, 21, 24, 54, 34\}$ berechnen. Die Ordnung P_F ist links unten im Bild dargestellt.

Lemma 18. *Wenn kein Block B der Partition mehr durch ein Splitelement $s \notin B$ zerlegt werden kann, so folgt: Jeder Block der Partition ist Modul.*

Beweis. Sei B Block, $s \notin B$; wenn B nach $\textsc{Split}(s)$ noch Block der Partition ist, dann ist $B = B^*$ oder $B = B_*$. Ist B unzerlegbar, so verhalten sich also alle $x \in B$ zu jedem $s \notin B$ gleich, d.h. B ist Modul. □

Aus dem Lemma ergibt sich unmittelbar die

Folgerung 19. Wenn der Graph G ein primer Vergleichbarkeitsgraph ist und $c = \textsc{CandSplit}(G, v)$ dann kann die Anfangspartition $(V - c, c)$ durch Split-Operationen solange verfeinert werden, bis jeder Block der Partition einelementig ist. Also ist die zuletzt erreichte geordnete Partition eine lineare Erweiterung der transitiven Orientierung F von G. □

Unser nächstes Ziel ist es aus diesem Ansatz einen $O(m \log n)$-Algorithmus zu machen. Dazu genügt es, wenn die folgenden zwei Bedingungen erfüllt werden können.

(1) Der Aufwand für $\textsc{Split}(s)$ kann auf $\mathcal{O}(|N(s)|)$ Zeit beschränkt werden.

(2) Jedes Element wird höchstens $\mathcal{O}(\log n)$-mal als Splitelement verwendet.

Dann nämlich ist $\sum_{v \in V} \log n \cdot d_v = \log n \sum_{v \in V} d_v = 2m \log n$.

Ad (1) Wir stellen den Graphen G als einen Array von Knoten dar, die wiederum als Records deklariert werden:

type
Node = **record**
    neighbourlist: **pointer to** ListNode
    positionInBlocklist: **pointer to** BlockListNode

Dabei deklarieren wir weiterhin:

ListNode = **record**
    identity: $\{1 \,..\, n\}$
    next: **pointer to** ListNode **or nil**
    previous: **pointer to** ListNode **or nil**
BlockListNode = **record**
    identity: $\{1 \,..\, n\}$
    next: **pointer to** BlockListNode **or nil**
    previous: **pointer to** BlockListNode **or nil**
    blockaddress: **pointer to** Block **or nil**

Die geordnete Partition stellen wir dar, indem wir ihre Blöcke ebenfalls als Records deklarieren:

```
Block = record
        first: {1 .. n}
        last: {1 .. n}
        helplength: {0 .. length − 1}
        nextBlock: pointer to Block or nil
        previousBlock: pointer to Block or nil
        blocklist: pointer to BlockListNode
        helplist: pointer to BlockListNode or nil
```

Außerdem benötigen wir eine Liste der Blöcke, die während der Split-Operation mit Element s angefaßt werden und die geordnete Partition als Liste von Blöcken:

```
var
touchedList: pointer to Block or nil
Partition: pointer to Block or nil
```

Wenn wir nun während eines Splits die Nachbarschaftsliste eines Knotens s durchlaufen und wissen wollen, an welcher Position der zugehörigen Blockliste B sich ein Nachbarschaftslistenknoten x befindet, so finden wir diese Information im Feld $G[x.\text{identity}].\text{positionInBlocklist}$. Diese Knoten werden dann in die Hilfsliste B^* $(= B \cap N(s))$ eingetragen. Die verbleibende Haupt-Blockliste B_* $(= B \backslash B^*)$ wird dann zur neuen Blockliste B. Dieses Eintragen der Nachbarn von s in die Hilfslisten ihrer Blöcke benötigt nur $\mathcal{O}(d_s)$ Zeit.

Anhand der Werte B.first und B.last kann entschieden werden, ob B vor oder hinter s liegt. Wir können dann die Anfangs- und Endwerte für B und B^* reparieren ($\mathcal{O}(d_s)$ Zeit) und für Elemente x aus B^* die Pointer von x auf Block(x) neu setzen (ebenfalls $\mathcal{O}(d_s)$ Zeit).

Diese Ideen sind die Grundlage für den in der Abbildung widergegebenen detailierten Code für Split. Der Code verwendet die oben definierten Datentypen und darüberhinaus einige einfache Listenoperationen wie *delete* und *insert_before*.

Folgerung 20. Die Zeitkomplexität für Split(s) kann auf $O(d_s)$ beschränkt werden. □

Als hinreichend für eine Gesamtlaufzeit von $O(m \log n)$ hatten wir oben zwei Bedingungen formuliert. Die Erfüllbarkeit der ersten ist obige Folgerung. Die zweite Bedingung war: Jedes Element wird höchstens $\mathcal{O}(\log n)$-mal als

```
SPLIT(s)
    Bs ← G[s].positionInBlocklist.blockaddress
    for all x in G[s].neighbourlist do
        bx ← G[x.identity].positionInBlocklist
        Bx ← bx.blockaddress
        if Bx ≠ Bs do
            delete( bx, Bx.blocklist)
            headinsert( bx, Bx.helplist)
            if Bx.helplength = 0 do headinsert( Bx, touchedList)
            Bx.helplength++
    for all B in touchedList do
        new Block B*
        B*.blocklist ← B.helplist
        B.helplist ← nil
        for all bx in B*.blocklist do
            bx.blockaddress ← address(B*)
        if B.last < Bs.first do
            B*.first ← B.first
            B*.last ← B*.first + B.helplength − 1
            B.first ← B*.last + 1
            insert_before( B*, B, Partition )
        if B.fist > Bs.last do
            B*.last ← B.last
            B*.first ← B*.last − B.helplength + 1
            B.last ← B*.first − 1
            insert_after( B*, B, Partition )
        B.helplength ← 0
```

Abbildung 2: Verfeinerter Code für SPLIT

Splitelement verwendet. Die Idee um dieses Ziel zu erreichen ist, daß wir für jeden Block einen Zähler *free* verwenden, der angibt, wie klein der Block werden muß, bevor wir seine Elemente wieder als Splitelemente verwenden werden.

NodePartit(G, c)

$L \leftarrow [V - c \mid c\,]$
free$(V - c) \leftarrow \frac{n}{2}$
free$(c) \leftarrow 2$
while $\exists X$ **with** free$(X) \geq |X|$
  free$(X) \leftarrow \frac{1}{2}|X|$
  for all $s \in X$
    for all $B \neq X$ in L
      $B^* \leftarrow B \cap N(s)$
      $B \leftarrow B \backslash B^*$
      free$(B^*) \leftarrow$ free(B)
      if B before X in L
        replace B by B^*B
      else
        replace B by BB^*
return L

Lemma 21. *Ein Element wird von* NodePartit *höchstens* $(\log n)$*-mal als Splitelement benutzt.*

Beweis. $|X_1| \leq n$, $|X_2| \leq \frac{|X_1|}{2}$, ..., $|X_t| \leq \frac{|X_{t-1}|}{2}$, und somit ist $n \geq 2^t$. □

Lemma 22. *Der größte Block B in der Rückgabeliste L ist ein Modul.*

Beweis. Sei $z \in (V - c) \setminus B$ und D der (zeitlich) letzte Block, in dem B und z gemeinsam enthalten waren. Mit B_z bezeichnen wir den augenblicklichen Block von z. Dann ist $|B_z| \leq \frac{1}{2}|D|$. Daher wurden die Elemente von B_z, insbesondere z selbst, nach dem Zerfall von D als Splitelemente verwendet. Also verhalten sich alle Elemente von B zu jedem $z \in (V - c) \setminus B$ gleich. Zu $z = c$ verhalten sich alle Elemente von B gleich, weil c das erste Splitelement war. Also ist B ein Modul. □

Satz 23. *Sei G primer Vergleichbarkeitsgraph. Dann ist die geordnete Partition $L :=$ NodePartit(G,CandSplit(G, v)) die lineare Erweiterung einer transitiven Orientierung F von G. Die Laufzeit ist $O(m \log n)$.*

Beweis. Das Element $c =$CandSplit(G, v)) kann in $O(m + n)$ berechnet werden und ist Maximum in einer transitiven Orientierung F von G (Lemma 15 und Lemma 16). Da NodePartit einen primen Graphen durch Split-Operationen in einelementige Blöcke zerlegt (Lemma 22) folgt aus Folgerung 19, daß die Partition L lineare Erweiterung von F ist. Die Erfüllbarkeit der beiden Bedingungen die eine Zeitkomplexität von $O(m \log n)$ gewährleisten, wurde mit Folgerung 20 und Lemma 21 nachgewiesen. □

Satz 24. *Für zusammenhängende Graphen G gilt:*
G prim $\Longleftrightarrow$ CandSplit(G, v) *gibt ein Element c zurück*
NodePartit(G, c) *enthält nur einelementige Blöcke*

Beweis. Die Implikation ($\Rightarrow$) ist durch den Beweis von Satz 15 (wenn CandSplit kein c zurückgibt dann gibt es ein nichttriviales Modul) und Lemma 22 abgedeckt.

Umgekehrt, sei G nicht prim, es ist zu zeigen, daß eine der Bedingungen nicht erfüllt ist. Wenn M ein Modul in $V\backslash\{c\}$ ist, dann bleibt M bei allen Split-Operationen erhalten und ist in einem Block der Rückgabeliste enthalten. Also dürfen wir annehmen, daß c in allen nichttrivialen Modulen enthalten ist.

Ein Modul M mit $c \in M$ und $v \notin M$ bleibt während des Laufs von CandSplit in C enthalten, also kann der Al;gorithmus nicht ein einzelner Element zurückgeben – Widerspruch. Daher sind v und c gemeinsam in allen nichttrivialen Modulen enthalten. Sei M das kleinste solche Modul. Sei A die Menge der Nachbarn der Elemente von M in $V - M$ und B die Menge der Nichtnachbarn. Angenommen $B = V - M$ oder $A = V - M$, in beiden Fällen ist $V - M$ ein Modul.

- Ist $|V - M| = 1$ dann ist $A = V - M$ weil G zusammenhängend und wenn $A = \{a\}$ dann ist $c \neq a =$CandSplit(G, v), Widerspruch.
- Ist $|V - M| > 1$ dann ist $V - M$ ein nichttriviales Modul das c nicht enthält, Widerspruch.

Also ist $A \neq \emptyset$ und $B \neq \emptyset$. Nach der ersten Runde in der $C = V - v$ mit v gesplitet wurde ist $C \subseteq A \cup M$ und $B \subseteq Q$. Nachdem mit einem Element $b \in B$ gesplitet wurde ist $C \subseteq A$, im Widerspruch zu $c \in M$ und $c =$CANDSPLIT(G, v). □

Unser Ziel war einen Erkennungsalgorithmus für Vergleichbarkeitsgraphen zu entwickeln. Die Überlegungen in den Abschnitten 9.2 und 9.3 legten nahe, daß wir uns auf das einfachere Problem der Erkennung primer Vergleichbarkeitsgraphen beschränken. Satz 24 beweist, daß wir prime Graphen erkennen können. Ein Graph ist ein Vergleichbarkeitsgraph, genau wenn er eine transitive Orientierung besitzt. Satz 23 zeigt wie wir für prime Vergleichbarkeitsgraphen eine transitive Orientierung berechnen können. Für die Erkennung von primen Vergleichbarkeitsgraphen fehlt nun noch die Verifikation, daß eine gegebene Orientierung transitiv ist. In anderen Worten, was dann noch fehlt, ist ein Algorithmus für das Problem: Gegeben eine Orientierung A von G. Entscheide, ob A transitiv ist.

Es ist einfach einen $O(n^3)$ Algorithmus für das Problem zu entwerfen: A ist nicht transitiv genau wenn es (x, y, z) gibt mit $(x, y) \in A$ und $(y, z) \in A$ aber $(x, z) \notin A$. Also prüft man alle Tripel (x, y, z).

Übung 4. *Das Problem kann auf Matrixmultiplikation zurückgeführt werden.*

Folgerung. Das Problem kann in $O(n^{2{,}39\cdots})$ gelöst werden.

Beweis. Matrixmultiplikation kann in $O(n^{2{,}39\cdots})$ durchgeführt werden (für mehr zu diesem Thema siehe etwa [10]). □

Zum Kapitel

Die ursprüngliche Quelle zu Vergleichbarkeitsgraphen ist Gallai [16]. Algorithmisch wurde die Erkennung von Vergleichbarkeitsgraphen zuerst von Pnueli, Lempel und Even und anschließend von Golumbic behandelt. Diese Autoren verfolgen den naiven Ansatz, orientiere eine Kante und erzeuge die zugehörige Implikationsklasse. Eine gründliche Darstellung eines $O(n^3)$ Algorithmus der auf dieser Idee basiert gibt Golumbic [18].

Substitutionsdekomposition für Graphen ist ein Spezialfall einer allgemeineren Dekomposition die für diskrete Strukturen. Eine ausführliche Darstellung die auch verschiedene algorithmische Anwendungen anspricht geben Möhring und Radermacher [39]. Einen Überblick über die Zeitschranken für Dekompositionsalgorithmen haben wir schon gegeben, genaue Literaturangaben finden sich bei McConnel und Spinrad [33].

Für transitive Orientierung wurden $O(n^2)$ Algorithmen von Spinrad entwickelt. Die in diesem Kapitel vorgestellten Algorithmen basiert auf dem Konferenzbeitrag [33] von 1994. Zuletzt konnten McConnel und Spinrad auch transitive Orientierung auf Linearzeit drücken [34]. An der Vereinfachung der Ergebnisse von McConnel und Spinrad wird noch zu Arbeiten sein.

10 Mehr über Ordnungen

10.1 Dimension

22. Vorlesung, 8.2.1996
Mitschrift: Helmut Weil

Zunächst wollen wir einen Beweis für das folgende Lemma geben, (vergl., Übung 8 in Kapitel 2).

Lemma 1. *Jede Ordnung besitzt eine lineare Erweiterung.*

Beweis. Sei $P = (X, <)$ eine Ordnung. Wir geben einen Algorithmus zur Erzeugung einer linearen Erweiterung L von P an. Dabei bezeichne $P[X']$ für $X' \subseteq X$ die auf X' eingeschränkte Ordnung P und $Min(P)$ die Menge der minimalen Elemente in P.

LinExt(P)
  $L \leftarrow [\,]$
  while $X \neq \emptyset$
    choose $x \in \mathrm{Min}(P[X])$
    $L \leftarrow L + x$
    $X \leftarrow X - x$
  return L □

LinExt ist ein generischer Algorithmus. Durch geeignete Spezifikation des Auswahlverfahrens *choose* kann man jede lineare Erweiterung von P durch diesen Algorithmus erzeugen lassen. Wir werden den Algorithmus im folgenden dazu verwenden, uns lineare Erweiterungen mit bestimmten Eigenschaften zu verschaffen.

Doch zuerst brauchen wir noch einige Definitionen: Wenn $P = (X, <_P)$ und $Q = (X, <_Q)$ zwei Ordnungen auf der gleichen Menge X sind, dann definieren wir die Ordnung $P \cap Q$ als Durchschnitt der Mengen der geordneten Paare der Relationen $<_P$ und $<_Q$, also

$$x <_{P\cap Q} y \quad \Longleftrightarrow \quad x <_P y \,\wedge\, x <_Q y$$

Mit $\mathcal{L}(P)$ bezeichnen wir die Menge aller linearen Erweiterungen von P.

Lemma 2. *Jede Ordnung ist Durchschnitt ihrer linearen Erweiterungen.*

$$P \;=\; \bigcap_{L \in \mathcal{L}(P)} L$$

Beweis. Für eine Ordnung $P = (X, <_P)$ gilt $P \subseteq \bigcap_{L\in\mathcal{L}(P)} L$, weil für alle $L \in \mathcal{L}(P)$ jede Relation in P auch Relation in L ist. Sei jetzt $z \in X$ und $L_z :=$LinExt(P), wobei *choose* jedes Element $x \neq z$ gegenüber z bevorzugt, d.h. in L_z kommen hinter z nur noch solche $y \in X$ vor, für die $z <_P y$ gilt. Dann gilt für $x, y \in X$ mit $x \,\|\, y$ (unvergleichbar in P), x ist vor y in L_y und y ist vor x in L_x, also $x \,\|\, y$ in $L_x \cap L_y$. Damit folgt

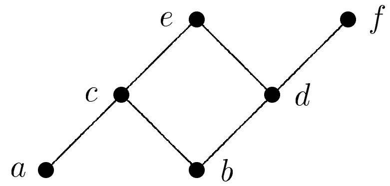
$$\bigcap_{L\in\mathcal{L}(P)} L \;\subseteq\; \bigcap_{z\in X} L_z \;\subseteq\; P$$

$\square$

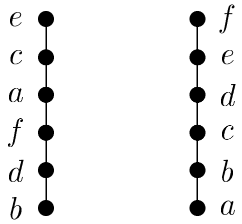
Eine Menge linearer Erweiterungen $\mathcal{R} \subseteq \mathcal{L}(P)$ heißt *Realizer* von P, falls $\bigcap_{L\in\mathcal{R}} L = P$ gilt. Die Dimension der Ordnung P definieren wir als die Größe eines minimalen Realizers von P, also $\dim(P) := \min(|\mathcal{R}|)$, wobei $\mathcal{R}$ Realizer von P. Da nach obigem Lemma $\mathcal{L}(P)$ ein Realizer von P ist, ist die Dimension wohldefiniert. Aus dem Beweis bekommen wir auch eine erste Schranke für die Dimension: $\dim(P) \le |X|$.

Beispiele.

(i) Eine Ordnung $P = L_1 \cap L_2$.

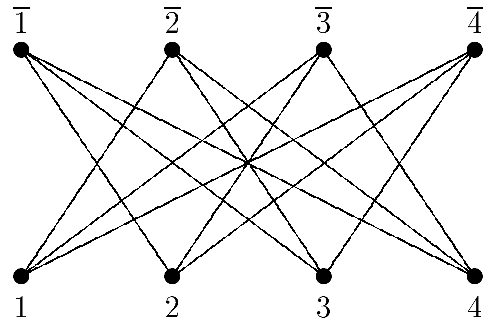


P mit $\dim(P) = 2$



L_1 L_2

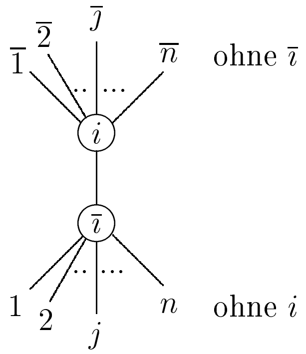
(ii) Mit S_n bezeichnen wir die durch die ein- und $(n-1)$-elementigen Teilmengen einer n-elementigen Menge induzierte Ordnung auf $2n$ Knoten, d.h. wenn $M := \{1, 2, \ldots, n\}$ die zugrundeliegende Menge ist und wir abkürzend $i := \{i\}$ und $\bar\imath := M - i$, $(i = 1, \ldots, n)$ für die Teilmengen schreiben, dann ist $S_n = (\{i : i \in M\} \cup \{\bar\imath : i \in M\}, \subset)$. Am Beispiel S_4 ist dies im nachfolgenden Bild illustriert.



Die Ordnung S_4

Behauptung. $\dim(S_n) \geq n$.

Beweis. In jedem Realizer von S_n muß es eine lineare Erweiterung geben, in der $\bar\imath$ vor i steht, aber in keiner linearen Erweiterung kann $\bar\imath$ vor i und $\bar\jmath$ vor j stehen $(i \neq j)$.



Denn wenn es ein solches $L \in \mathcal{L}(S_n)$ gäbe, dann wäre $\bar\imath <_L i <_{S_n} \bar\jmath <_L j <_{S_n} \bar\imath$ – Widerspruch. Ein Realizer von S_n hat also für jedes $i = 1, 2, \ldots, n$ eine lineare Erweiterung mit $\bar\imath$ vor i.

□

Schranken für die Dimension

Satz 3. *Für eine Ordnung $P = (X, <_P)$ mit Weite $w(P)$ gilt:*

$$\dim(P) \leq w(P)$$

Beweis. Sei $C_1, \ldots, C_w$ eine Zerlegung von P in $w = w(P)$ viele Ketten. Diese Zerlegung existiert nach dem Satz von Dilworth. Sei weiter für $i = 1, \ldots, w$ L_i =LinExt(P), wobei diesmal *choose* jedes Element $\notin C_i$ gegenüber jedem Element $\in C_i$ bevorzugt, d.h. daß C_i in L_i „so hoch wie möglich" liegt. Da die L_i lineare Erweiterungen sind ist $P \subseteq \bigcap_{i=1}^{w} L_i$ erfüllt. Für die andere Richtung seien $x, y \in X$ in P unvergleichbare Elemente, dann sind sie in verschiedenen Ketten, d.h. $x \in C_i, y \in C_j$ und $i \neq j$. Man überzeugt sich schnell, daßt $x <_{L_j} y$ und $y <_{L_i} x$. Also sind x und y unvergleichbar in $L_i \cap L_j$.

$\square$

Satz 4. *Sei $P = (X, <_P)$ eine Ordnung und $x \in X$, dann gilt:*

$$\dim(P) - 1 \leq \dim(P - x) \leq \dim(P)$$

Beweis. Wir wollen zuerst die zweite Ungleichung $\dim(P - x) \leq \dim(P)$ zeigen. Dazu sei $\{L_1, \ldots, L_r\}$ ein Realizer von P und $L_i^* := L_i - x$ für $i = 1, \ldots, r$. (Dabei soll $L_i - x = L_i[X - x]$ die auf $X - x$ eingeschränkte Ordnung L_i bezeichnen.) Dann ist $\{L_1^*, \ldots, L_r^*\}$ ein Realizer von $P - x$.

Zum Beweis der ersten Ungleichung führen wir erst noch eine Abkürzung ein. Die Menge $\{y \in X : x <_P y\}$ heißt *Upset* von x und wird mit $U(x)$ bezeichnet. Analog ist $D(x) = \{y \in X : y <_P x\}$ das *Downset* von x.

Sei $\{L_1^*, \ldots, L_r^*\}$ ein Realizer von $P - x$. Dann gibt es in jedem L_i^* eine Stelle, an der x eingefügt werden kann, so daß $L_i = L_i^* + x$ eine lineare Erweiterung von P ist. Eine solche Stelle ist z.B. unmittelbar vor dem $y \in U(x)$, das in L_i^* am weitesten links steht. Denn wenn wir annehmen, das wäre nicht der Fall, d.h. das so erzeugte L_i wäre keine lineare Erweiterung von P, dann muß eine Relation mit x in P verletzt sein. Weil für alle $y \in U(x)$ nach Konstruktion $x <_{L_i} y$ gilt, bleibt nur übrig, daß es ein $z \in D(x)$ gibt mit $x <_{L_i} z$. Dann ist z in L_i^* hinter dem am weitesten links stehenden Element $y \in U(x)$ und es folgt wegen $z <_P x <_P y$ auch $z <_{L_i} x <_{L_i} y <_{L_i} z$, was unmöglich ist.

Wir können also x in $L_2^*, \ldots, L_r^*$ einfügen und erhalten lineare Erweiterungen $L_2, \ldots, L_r$ von P. Dann bilden wir L_0 und L_1 folgendermaßen:

$$L_0 = L_1^*[D(x)] \ , \ x \ , \ L_1^*[X - D(x)]$$

$$L_1 = L_1^*[X - U(x)] \ , \ x \ , \ L_1^*[U(x)]$$

In L_0 liegen also genau alle $y \in D(x)$ in der Reihenfolge wie in L_1^* vor x, und alle anderen dahinter, in L_1 liegen genau alle $y \in U(x)$ in der Reihenfolge wie in L_1^* hinter x, alle anderen davor. Anders ausgedrückt: mit x unvergleichbare Elemente sind weder in $D(x)$ noch in $U(x)$, liegen also in L_0 hinter x und in L_1 vor x. Außerdem bleiben alle Relationen, an denen x nicht beteiligt ist, erhalten.

Insgesamt folgt, daß $\{L_0, \ldots, L_r\}$ ein Realizer von P ist und damit die erste Ungleichung. $\square$

Satz 5. *Für jede Ordnung* $P = (X, <)$ *gilt:* $\dim(P) \leq \frac{|X|}{2}$.

Bemerkung. Wir wollen das Theorem hier nicht beweisen, sondern nur auf folgendes hinweisen: Es scheint naheliegend, zu versuchen den Beweis per Induktion zu führen. Dazu wünscht man sich ähnlich wie im vorletzten Satz eine Aussage der Form

$$\forall P\ \exists x, y \in X : \quad \dim(P - \{x, y\}) \geq \dim(P) - 1.$$

Das ist allerdings eine bisher *unbewiesene* Vermutung. (Der Beweis des Satzes geht also anders.)

Dominanzordnung im $\mathbb{R}^n$

Wir wollen uns jetzt der Dimension von Ordnungen aus einer anderen Richtung nähern. Für Punkte $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n)$ des $\mathbb{R}^n$ sagen wir x *dominiert* y gdw. $x_i \geq y_i$ für alle $i = 1, \ldots, n$, und das definiert uns eine Ordnung der Punkte des $\mathbb{R}^n$, die *Dominanzordnung*.

Wenn eine Ordnung $P = (X, <_P)$ mittels einer Abb. $\phi : X \to \mathbb{R}^k$ so in den $\mathbb{R}^k$ eingebettet werden kann, daß

$$\forall x, y \in X : \ x <_P y \quad \Longleftrightarrow \quad \phi(y) \text{ dominiert } \phi(x)$$

dann sagen wir $\dim_{\mathcal{O}}(P) \leq k$. Wenn k die kleinste Zahl ist, so daß sich P auf diese Weise in den $\mathbb{R}^k$ einbetten läßt, dann ist $\dim_{\mathcal{O}}(P) = k$. Später werden wir, wenn keine Verwechslungen zu befürchten sind, die Abb. ϕ meist weglassen und x mit $\phi(x)$ identifizieren.

Beobachtung. $\dim(P) = \dim_{\mathcal{O}}(P)$.

Beweis. Sei $P = (X, <_P)$ eine Ordnung und $\{L_1, \ldots, L_k\}$ ein minimaler Realizer von P. Dann definieren wir eine Abbildung

$\phi = (\phi_1, \ldots, \phi_k) : X \to \mathbb{R}^k$ wobei $\phi_i(x)$ die Position von x in L_i ist.

Dann gilt für alle $x, y \in X$:

$$\begin{array}{rccl} & \phi(y) \text{ dominiert } \phi(x) & \Leftrightarrow & \forall i \leq k : \phi_i(x) \leq \phi_i(y) \\ \Leftrightarrow & \text{in jedem } L_i \text{ liegt } x \text{ vor } y & \Leftrightarrow & x <_P y. \end{array}$$

Also ist $\dim(P) \leq \dim_{\mathcal{O}}(P)$.

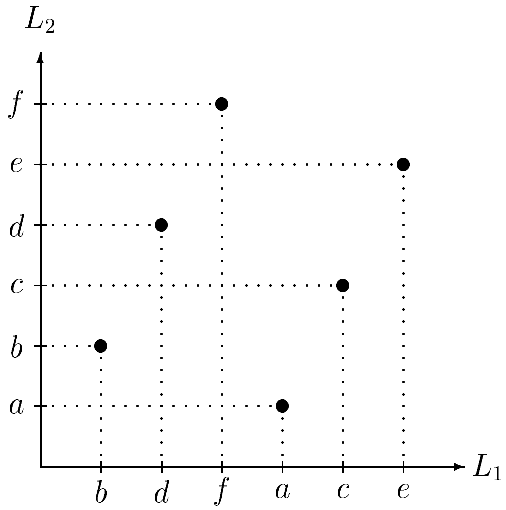
Sei umgekehrt $P = (X, <_P)$ eine mittels einer Abb. $\phi = (\phi_1, \ldots, \phi_k)$ in den $\mathbb{R}^k$ eingebettete Ordnung, dann definieren wir für $i = 1, \ldots, k$ L_i als die nach den i-ten Koordinaten sortierte Liste der Punkte aus $\phi(X)$, d.h. $x <_{L_i} y$ gdw. $\phi_i(x) < \phi_i(y)$.

Dann gilt für alle $x, y \in X$:

$$\begin{array}{rccl} & \forall i \leq k : x <_{L_i} y & \Leftrightarrow & \forall i \leq k : \phi_i(x) < \phi_i(y) \\ \Leftrightarrow & \phi(y) \text{ dominiert } \phi(x) & \Leftrightarrow & x <_P y. \end{array}$$

Also ist $\dim_{\mathcal{O}}(P) \leq \dim(P)$. □

Beispiel. Zur Illustration des Beweises betrachten wir noch einmal unser P von früher, diesmal eingebettet in $\mathbb{R}^2$ mit der Dominanzordnung.



Einbettung von P im $\mathbb{R}^2$

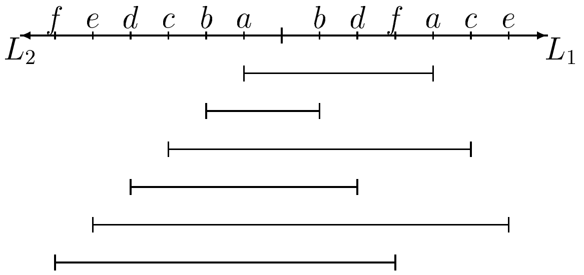
Bemerkung. Die Entscheidung, ob eine gegebene Ordnung eine Dimension $\leq k$ hat, ist NP-vollständig für $k \geq 3$.

10.2 2-dimensionale Ordnungen

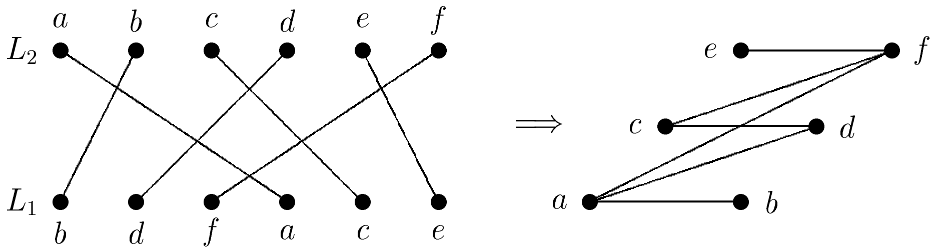
Zuerst wollen wir einige Eigenschaften 2-dimensionaler Ordnungen am Beispiel illustrieren.

Beispiel. Aus der Einbettung unserer Beispielordnung P können wir eine Inklusionsordnung von Intervallen erzeugen, indem wir die Achsen L_2 und L_1 wie im nächsten Bild unten „aufklappen“ und Intervalle I_x mit den Endpunkten $x \in L_2$ und $x \in L_1$ für alle x definieren. Dann gilt (nicht nur in diesem Beispiel, sondern allgemein für 2-dim. Ordnungen)

$$I_x \subset I_y \quad \Leftrightarrow \quad x <_P y$$



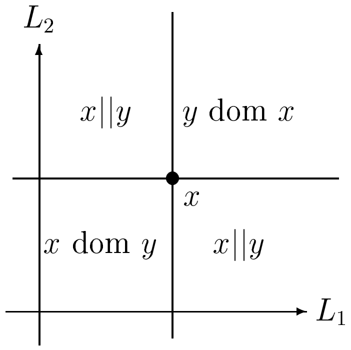
Wenn wir L_1 und L_2 übereinanderlegen und die Positionen der selben Knoten verbinden, definiert uns das einen *Permutationsgraphen* (vergl Beispiel 2 aus Abschnitt 1.2). Zwei Kanten schneiden sich gdw. die zugehörigen Elemente in P unvergleichbar sind.

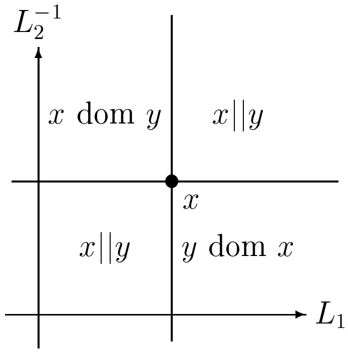


Permutationsgraphen sind die Unvergleichbarkeitsgraphen 2-dimensionaler Ordnungen.

Lemma 6. *Wenn P eine Ordnung der Dimension 2 ist und G der Vergleichbarkeitsgraph von P, dann ist $\overline{G}$ auch ein Vergleichbarkeitsgraph.*

Beweis. Sei $\{L_1, L_2\}$ ein Realizer von P, d.h. $P = L_1 \cap L_2$. Dann wird für jedes $x \in X$ durch die Dominanzordnung der $\mathbb{R}^2$ in 4 Quadranten unterteilt, von denen je zwei gegenüberliegende nur mit x bzgl. der Dominanzordnung vergleichbare und die anderen beiden nur mit x unvergleichbare Elemente y enthalten, wie im Bild unten links angedeutet ist. Wenn wir jetzt das Bild an der L_1-Achse spiegeln, bzw. wie im rechten Bild die L_2-Achse durch L_2^{-1} ersetzen (umdrehen), dann erhalten wir eine neue Ordnung $P' = L_1 \cap L_2^{-1}$.



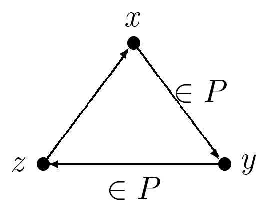


Dann ist jedes Paar x, y entweder in P oder in P' vergleichbar. Daher gilt wenn G der Vergleichbarkeitsgraph von P und G' der Vergleichbarkeitsgraph von P' ist, daß $G' = \overline{G}$ und damit ist gezeigt, daß $\overline{G}$ Vergleichbarkeitsgraph ist. □

Das diese Eigenschaft schon eine Charakterisierung der Vergleichbarkeitsgraphen 2-dimensionaler Ordnungen darstellt zeigen wir im folgenden Satz.

Satz 7. *P ist 2-dimensionale Ordnung genau dann, wenn der Unvergleichbarkeitsgraph $\overline{G}$ von P ein Vergleichbarkeitsgraph ist.*

Beweis. Wir müssen nur noch die Rückrichtung ($\Longleftarrow$) zeigen. Sei also G der Vergleichbarkeitsgraph von P, und Q transitive Orientierung von $\overline{G}$ (der ja nach Vorr. auch Vergleichbarkeitsgraph ist). Dann ist $L = P \cup Q$ eine Orientierung des K_n (wobei $n = |V(G)|$), und diese Orientierung ist transitiv bzw. azyklisch. Denn wenn sie das nicht wäre, enthielte sie einen gerichteten Kreis und damit auch ein gerichtetes Dreieck,von dessen drei Kanten entweder genau zwei $\in P$ oder genau zwei $\in Q$ sind. Wir nehmen o.B.d.A. $(x, y), (y, z) \in P$ an (siehe Bild).



Aus der Transitivität von P folgt aber $(x, z) \in P$, und damit $(z, x) \notin P \cup Q$, also haben wir einen Widerspruch und $P \cup Q$ ist azyklisch. Ganz analog folgt, daß $L' = P \cup Q^{-1}$ azyklische Orientierung des K_n ist und weil $P \cap Q = P \cap Q^{-1} = \emptyset$ und damit $P = (P \cup Q) \cap (P \cup Q^{-1}) = L \cap L'$ folgt: $\{L, L'\}$ ist Realizer von P, also $\dim(P) \leq 2$. $\square$

Erkennung 2-dimensionaler Ordnungen

Der Beweis des letzten Satzes ermöglicht uns, zum Schluß noch kurz ein Verfahren anzugeben zur Lösung des Problems der Erkennung 2-dimensionaler Ordnungen, bzw. genauer der Beantwortung der Frage:

> Ist ein gegebener primer Graph G Vergleichbarkeitsgraph einer Ordnung der Dimension 2 ?

Die Beschränkung auf prime Graphen rechtfertigt sich durch die Tatsache, daß wenn alle Module eines Graphen Vergleichbarkeitsgraphen 2-dimensionaler Ordnungen sind, dasselbe klarerweise auch für den ganzen Graphen gilt und umgekehrt.

Mit den Bezeichnungen des letzten Beweises sieht das Verfahren so aus:

Berechne zuerst $A = \text{NODEPARTIT}(G, c)$ und $B = \text{NODEPARTIT}(\overline{G}, d)$. Wenn G der prime Vergleichbarkeitsgraph einer 2-dimensionalen Ordnung P ist, dann ist, nach dem im Abschnitt über transitive Orientierung bewiesenen, A eine lineare Erweiterung von P und B eine von Q. Mit Hilfe von A und B können die Knoten nun in $O(n \log n)$ Zeit so sortiert werden, daß sie in der Reihenfolge $L = P \cup Q$ sind, und genauso für $L' = P \cup Q^{-1}$. Ist das getan, dann bleibt zu prüfen, ob $P = L \cap L'$? Das ist in $O(m)$ Zeit möglich. Das Verfahren insgesamt ist also in $O(m + n \log n)$ Zeit durchführbar.

10.3 k-Perfektheit

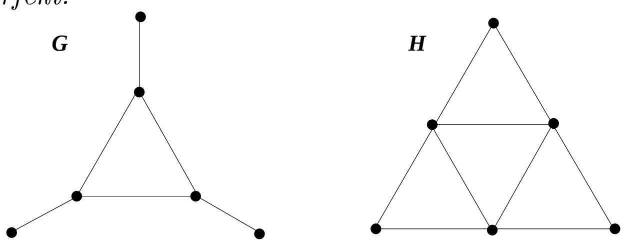
23. Vorlesung, 13.2.96
Mitschrift: Stefan Felsner

Die maximale Größe $\alpha(G)$ einer unabhängigen Menge ist in jedem Graphen G beschränkt durch die Größe eine Cliquenzerlegung $\mathcal{C}$. Sei $\alpha_k(G)$ die maximale Größe einer Vereinigung von k unabhängigen Mengen. Eine Menge die $\alpha_k(G)$ maximiert kann auch als maximum k-färbbare Teilmenge von V beschrieben werden. Jede Clique kann höchstens k Knoten mit so einer Menge gemeinsam haben, also liefern Cliquenzerlegungen auch Schranken für α_k. Diese Schranken hängen nur von den Größen der Cliquen ab, daher wir definieren wir für eine endliche Multimenge natürlicher Zahlen A die *k–Norm* als $m_k(A) = \sum_{a \in A} \min(k, a)$. Für eine Cliquenzerlegung $\mathcal{C}$ schreiben wir abkürzend $m_k(\mathcal{C})$ wenn wir die k–Norm der Größen von Cliquen in $\mathcal{C}$ meinen. Damit gilt also für jedes $k \in \mathbb{N}$ und jede Cliquenzerlegung $\mathcal{C}$

$$\alpha_k(G) \leq m_k(\mathcal{C}).$$

Ein Graph G heißt *k-Perfekt* wenn er für jedes k eine Cliquenzerlegung $\mathcal{C}$ besitzt, so daß $\alpha_k(G) = m_k(\mathcal{C})$.

Übung 1. *Seien G und H wie im folgenden Bild. Zeige: G ist k-perfekt und H ist nicht k-perfekt.*



Da die Graphen G und H aus der Übung komplementär sind gilt das k Analogon des PG-Theorems im allgemeinen nicht. Als Übung in Matchingtheorie zeigt man:

Übung 2. *Bipartite Graphen und ihre Komplemente sind k-perfekt.*

Es gilt auch: Line-Graphen von bipartiten Graphen sind k-perfekt. Die Komplemente dieser Graphen haben die Eigenschaft aber, wie obiges Beispiel zeigt, schon nicht mehr.

Die k-Perfektheit von Vergleichbarkeitsgraphen folgt aus der folgenden Verallgemeinerung des Satzes von Dilworth.

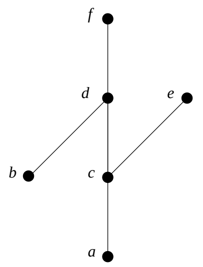
Satz 8. [Greene & Kleitman] *Für jede Ordnung P und jedes k gilt*

$$\max_{\mathcal{A}} |\bigcup_{A\in\mathcal{A}} A| = \min_{\mathcal{C}} m_k(\mathcal{C})$$

Dabei wird das Maximum über alle $\mathcal{A} = \{A_1, .., A_k\}$ mit A_i Antikette genommen und das Minimum über alle Kettenzerlegungen.

Es sei erwähnt, daß dies natürlich die k-Perfektheit bipartiter Graphen einschließt. Eine wichtige Beobachtung ist, daß das Minimum $m_k(\mathcal{C})$ für verschiedene k im allgemeinen verschiedenen Kettenzerlegungen erfordert.

Beispiel.



Die dargestellte Ordnung hat $\min_{\mathcal{C}} m_1(\mathcal{C}) = 2$ und $\min_{\mathcal{C}} m_3(\mathcal{C}) = 5$, der erste Wert wir ausschließlich von der Kettenzerlegung $\{a, c, e, \}\{b, d, f\}$ angenommen, der zweite von $\{a, c, d, f, \}\{b\}\{e\}$.

Es gilt allerdings das sogenannte *t-Phänomen*: Für alle t gibt es eine Kettenzerlegung $\mathcal{C}$ die sowohl $m_t(\mathcal{C})$ als auch $m_{t+1}(\mathcal{C})$ minimiert.

Die duale Aussage zu Satz 8 ist ebenfalls wahr.

Satz 9. [Greene] *Für jede Ordnung P und jedes ℓ gilt*

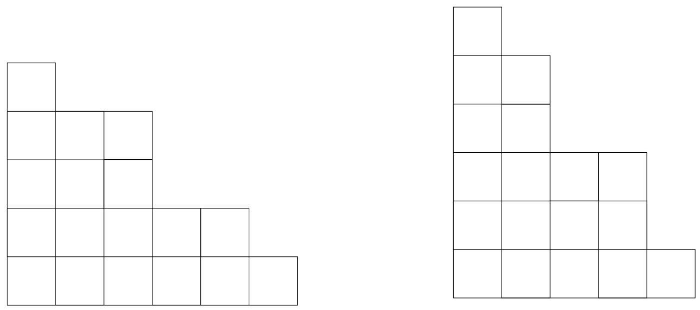
$$\max_{\mathcal{C}} |\bigcup_{C\in\mathcal{C}} C| = \min_{\mathcal{A}} m_\ell(\mathcal{A})$$

Dabei wird das Maximum über alle $\mathcal{C} = \{C_1, .., C_\ell\}$ mit C_i Kette genommen und das Minimum über alle Antikettenzerlegungen.

Wieder gilt das t-Phänomen. Für alle t gibt es eine Antikettenzerlegung $\mathcal{A}$ die sowohl $m_t(\mathcal{A})$ als auch $m_{t+1}(\mathcal{A})$ minimiert.

Eine *Partition* von $n \in \mathbb{N}$ ist eine Folge $\lambda_1 \geq \lambda_2 \geq .. \geq \lambda_p > 0$ mit $\lambda_i \in \mathbb{N}$ und $\sum_{i=1}^{p} \lambda_i = n$. Partitionen werden durch sogenannte *Ferrers Diagramme* graphisch dargestellt, siehe nachfolgendes Bild. Die i-te Zeile von unten im Ferrers Diagramm besteht aus λ_i Kästchen. Die *konjugierte Partition* von λ ist die Partition deren Ferrers Diagramm man aus dem Ferrers Diagramm von λ an der Diagonalen erhält.

Beispiel.



Die Ferrers Diagramme zur Partition $\lambda = (6, 5, 3, 3, 1)$ und der konjugierten $\mu = (5, 4, 4, 2, 2, 1)$ von λ.

Ein weiterer Satz von Greene (konjugierte Partitionen Satz) besagt, daß es zu jeder Ordnung P ein Ferrers Diagramm $g(P)$ gibt, für das gilt:

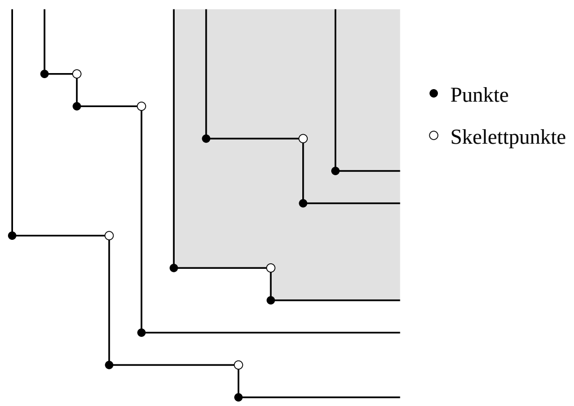
Satz 10. *Die Zahl der Kästchen in den k längsten Zeilen von $g(P)$ ist die maximale Größe einer Vereinigung von k Antiketten und die Zahl der Kästchen in den ℓ längsten Spalten von $g(P)$ ist die maximale Größe einer Vereinigung von ℓ Ketten, für alle $k, \ell \in \mathbb{N}$.*

10.4 Skelette 2-dimensionaler Ordnungen

Wir erinnern uns, daß eine 2-dimensionale Ordnung P eine Darstellung als Dominanzordnung von Punkten in der Ebene besitzt. Für einen Punkt $p = (x, y) \in \mathbb{R}^2$ sei der *Schatten* von p die Menge $\sigma(p) = \{q = (u, v) : u > x \text{ und } v > y\}$. Der Schatten einer Menge $Z \subset \mathbb{R}^2$ sei $\sigma(Z) = \bigcup_{p \in Z} \sigma(p)$. Der topologische Rand $\partial\sigma$ eines Schattens heißt *Sprunglinie*. Die Sprunglinie einer endlichen Antikette A hat runter/rechts-Knicke in den Punkten von A, die Punkte in den rechts/runter-Knicken der Sprunglinie heißen die *Skelettpunkte* $S(A)$ der Antikette. Es gilt $|S(A)| = |A| - 1$.

Die minimalen Element einer Menge X sind eine Antikette A, so daß der Rest $X - A$ im Schatten von A liegt. Entfent man A und behandelt den Rest wieder so, so erhält man die *kanonische Antikettenzerlegung* $A_1, .., A_\lambda$ die Antikette A_i wird auch die i-te *Lage* von X genannt. Die Sprunglinien $\partial\sigma(A_i)$ der Lagen von X schneiden sich nicht. Das *Skelett* $S(X)$ definieren wir als Vereinigung der Skelette der Lagen von X. Das folgende Bild zeigt eine Menge X, ihre kanonische Antikettenzerlegung, die Sprunglinien der Antiketten, das

Skelett $S(X)$ und schattiert den Schatten der 3-ten Lage von X.



Übung 3. *Sei $A_1, .., A_k$ eine Antikettenzerlegung von X, wenn die Sprunglinien $\partial\sigma(A_i)$ paarweise disjunkt sind, dann ist die Antikettenzerlegung die kanonische.*

Wir beschreiben nun einen Algorithmus zur Berechnung des Skeletts und einer minimum Antikettenzerlegung einer Punktmenge X. Wir stellen uns vor, daß der Algorithmus die Ebene von rechts nach links mit einer vertikalen Geraden (*sweep line*) L überstreicht. Auf der Geraden sind Markierungen die waagrechten Abschnitten von Sprunglinien entsprechen. Wenn L einen Punkt p aus X erreicht, dann suchen wir die nächste Sprunglinie (Markierung) oberhalb von p. Existiert so eine Sprunglinie, dann wird ein Skelettpunkt erzeugt, p wird der Lage der Sprunglinie zugeordnet, die alte die Sprunglinie repräsentierende Markierung gelöscht und eine neue auf der Höhe von p eingefügt. Wenn es noch keine Sprunglinie oberhalb von p gibt, dann eröffnet p eine neue Lage und eine Markierung auf der Höhe von p wird in L eingefügt.

Die Operationen die auf L ausgeführt werden müßen sind: Finde die nächste Markierung oberhalb von p, lösche eine Markierung und füge eine Markierung ein. Das sind typische Aufgaben für die spezielle Datenstrukturen entwickelt wurden (vergl citeCLR90). Verwendet man balancierte Suchbäume, dann kann jede der drei Aufgaben in $O(\log n)$ Zeit erfüllt werden, dabei ist n die Anzahl der in der Datenstruktur abgelegten Objekte, in unserem Fall also $n = |X|$.

SKELETT(X)
  $k \leftarrow 0$
  $S \leftarrow \emptyset$
  for each $p \in X$ from left to right
    insert m' in L with $(m'_y, m'_x) \leftarrow (p_y, L_x)$
    if $\exists$ m above p on L **then**
      add v to *antichain*(m)
      antichain(m') $\leftarrow$ *antichain*(m)
      add skeleton point (L_x, m_y) to S
      remove m from L
    else
      $k \leftarrow k + 1$
      $A_k \leftarrow \{p\}$
      antichain(m') $\leftarrow A_k$
  return $S, A_1, \ldots, A_k$

Satz 11. *Algorithmus* SKELETT *berechnet das Skelett und die kanonische Antikettenzerlegung von X mit $|X| = n$ in $O(n \log n)$ Zeit.* □

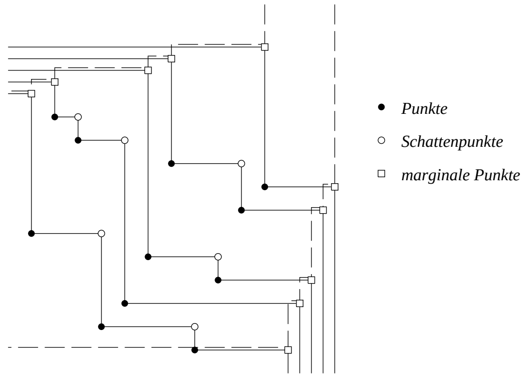
Übung 4. *Erweitere den Algorithmus so, daß er mit der Antikettenzerlegung auch eine maximum Kette berechnet.*

Sei $\sigma_{\swarrow}(p) = \{q = (u, v) : u < x \text{ und } v < y\}$ der Schatten von p wenn von rechts oben statt von links unten beleuchtet wird, wir wollen $\sigma_{\swarrow}(p)$ auch *dualen Schatten* nennen. Die Ränder dualer Schatten sind duale Sprunglinie., Die runter/rechts Knicke von dualen Sprunglinien sind duale Skelettpunkte. Die dualen Skelettpunkte der dual kanonischen Antikettenzerlegung bilden das duale Skelett $S_{\swarrow}(X)$.

Eine interessante Eigenschaft des Skeletts von X ist, daß wir aus $S(X)$ mit wenig zusätzlicher Information X zurückgewinnen können. Man benötigt zusätzlich zu $S(X)$ die x-Koordinaten der vertikalen Strahlen auf den Sprunglinien und die y-Koordinaten der horizontalen Strahlen auf den Sprunglinien. Den Prozess zur Zurückgewinnung von X beschreibt das folgende Lemma.

Lemma 12. *Seien $M_v(X)$ und $M_h(X)$ je eine Kette* marginaler Punkte *auf* *den vertikalen und auf den horizontalen Strahlen der Sprunglinien von X. Dann gilt $X = S_{\swarrow}(S(X) \cup M_v(X) \cup M_h(X))$.*

Die Konstruktion der marginalen Ketten und die Aussage des Lemmas veranschaulichen wir durch ein Bild.



Skelette und Young-Tableaux

Sei λ eine Partition von n, symbolisch $\lambda \vdash n$. Ein *Young-Tableau der Gestalt* λ besteht aus dem Ferrers Diagramms von λ, in dessen Kästchen die Zahlen $1, .., n$ so eingetragen sind, daß die Eintrage in jeder Zeile von links nach rechts und in jeder Spalte von unten nach oben aufsteigen.

Sei f_λ die Anzahl der Young-Tableaux der Gestalt λ. Es gilt die Bemerkenswerte Formel

$$n! = \sum_{\lambda \vdash n} f_\lambda^2.$$

Zum Beweis zeigt man

Satz 13. [Robinson-Schensted] *Es gibt eine Bijektion zwischen Permutationen $\pi \in \mathcal{S}_n$ und Paaren P_π, Q_π von Young-Tableaux derselben Gestalt.*

Vor dem Beweis des Satzes noch einige Definitionen und Vorüberlegungen. Für eine Punktmenge X sei $S^k(X)$ das *k-te Skelett von X*, also $S^k(X) = S(S^{k-1}(X)$ und $S^0(X) = X$. Mit $\mu(X)$ bezeichnen wir die kleinste Zahl für die $S^{\mu(X)+1}(X) = \emptyset$ gilt. Für $1 \leq i \leq \mu = \mu(X)$ sei λ_i die Anzahl der Lagen von $S^i(X)$.

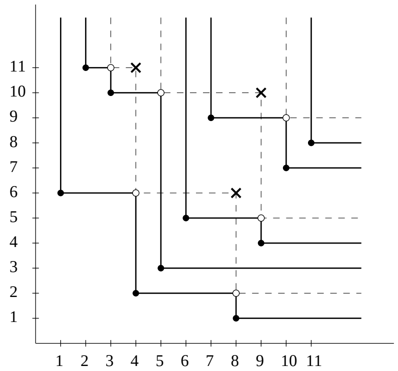
Lemma 14. *Sei X eine Punktmenge, dann gilt $\lambda_0 \geq \lambda_1 \geq ... \geq \lambda_\mu > 0$ und $|S^k(X)| = \sum_{k \leq i \leq \mu} \lambda_i$. Insbesondere ist $\lambda = (\lambda_0, \lambda_1, ..., \lambda_\mu)$ eine Partition von n.*

Beweis. λ_i die Anzahl der Lagen von $S^i(X)$ also die Anzahl der Antiketten einer minimalen Antikettenzerlegung. Die Sprunglinien von $S^i(X)$ induzieren eine Antikettenzerlegung der Skelettpunkte, also der Punkte von $S^{i+1}(X)$. Daher gilt $\lambda_{i+1} \leq \lambda_i$.

Aus $|S(A)| = |A| - 1$ für eine Antikette folgt $|X| = |S(X)| - \lambda_0$. Aus dieser Formel erhält man durch Induktion die $|S^k(X)| = \sum_{k \leq i \leq \mu} \lambda_i$. □

Beweis. (Des Satzes) Einer Permutation π sei die Punktemenge X_π zugeordnet, die aus den Punkten $(i, \pi(i))$ besteht. Das Young-Tableau P_π auch *P-Symbol* von π genannt erhält man wie folgt: Die k-te Zeile von P_π enthält die y-Koordinaten der horizontalen Strahlen der Sprunglinien von $S^k(X_\pi)$ in aufsteigender Reihenfolge. Die Konstruktion des Q-Symbols ist analog: Die k-te Zeile von Q_π enthält die x-Koordinaten der vertikalen Strahlen der Sprunglinien von $S^k(X_\pi)$ in aufsteigender Reihenfolge.

Das nächste Bild zeigt die ersten beiden Skelette von X_π wobei π die Permutetion $(6, 11, 10, 2, 3, 5, 9, 1, 4, 7, 8)$ ist. Man sieht, daß die ersten beiden Zeilen des P-Symbols durch $[1, 3, 4, 7, 8]$ und $[2, 5, 9]$ gegeben sind.



11				
6	10			
2	5	9		
1	3	4	7	8

8				
4	9			
3	5	10		
1	2	6	7	11

Das so konstruierte P-Symbol hat nach dem vorhergehenden Lemma die Gestalt einer Partition. Klar ist außerdem, daß die Gestalt des P und des Q-Symbols identisch ist. Auch das Wachstum der Einträge in den Zeilen der Symbole ist unmittelbar einzusehen.

Wir zeigen nun (für das Q-Symbol), daß in den Symbolen jeder Wert i aus $1, .., n$ vorkommmt: Betrachte den Punkt $(i, \pi(i))$, über diesem Punkt ist ein Stück Sprunglinie, entweder das ist ein vertikaler Strahl was bedeutet, daß i in der untersten Zeile des Q-Symbols vorkommt oder das Stück Sprunglinie knickt in einem Skelettpunkt ab. Man Iteriert die Überlegung bis ein vertikaler Strahl oberhalb von $(i, \pi(i))$ gefunden ist.

Das Wachstum der Einträge in den Spalten der Symbole folgt aus der Behauptung: Die i-te Sprunglinie von $S^{k+1}(X)$ liegt ganz im Schatten der i-ten Lage von $S^k(X)$.

Zum Beweis der Behauptung verfeinern wir das schon im Lemma verwendete Argument ein wenig. Die Sprunglinien von $S^k(X)$ induzieren eine Antikettenzerlegung $\{B_1, , .., B_{\lambda_k}\}$ von $S^{k+1}(X)$. Für $p \in B_i$ gilt, alle von p dominierten Punkte sind in Antiketten B_j mit $j < i$, anders gesagt die Höhe von Elementen in B_i ist höchstens i. Die Punkte der i-ten Sprunglinie von $S^{k+1}(X)$ sind genau die Punkte mit Höhe i, liegen also entweder auf der i-ten Sprunglinie von $S^k(X)$ oder auf einer Sprunglinie die im Schatten der i-ten Sprunglinie von $S^k(X)$ liegt.

Bisher haben wir uns davon überzeugt, daß die Konstruktion eine Abbildung von Permutationen auf Paare von Young Tableaux derselben Gestalt ist. Daß diese Abbildung eine Bijektion ist folgt im wesentlichen aus Lemma 12. Die Einträge in den k-ten Zeilen der Symbole sind genau die Information die wir benötigen um die marginalen Ketten $M_v(S^k(X))$ und $M_h(S^k(X))$ konstruieren zu können. Damit können wir dann $S^{k-1}(X)$ zurückerhalten. $\square$

Sei π^{-1} die Inverse der Permutation π, die Punktmenge $X_{\pi^{-1}}$ besteht aus den Punkten $(\pi(i), i)$, ist also die Spiegelung von X_π an der Diagonalen.

Folgerung. $P_{\pi^{-1}} = Q_\pi$ und $Q_{\pi^{-1}} = P_\pi$.

Wir nennen eine Menge von Elementen einer Ordnung die von k Antiketten überdeckt werden kann eine *k-Antikette*, analog ist eine *ℓ-Kette* eine Menge die von ℓ-Ketten überdeckt werden kann.

Satz 15. *Die Gestalt des P-Symbols von π ist das aus Satz 10 bekannte Greene Diagramm von X_π, d.h. die Zahl der Kästchen in den k längsten Zeilen ist die maximale Größe k-Antikette und die Zahl der Kästchen in den ℓ längsten Spalten ist die maximale Größe ℓ-Kette in X_π.*

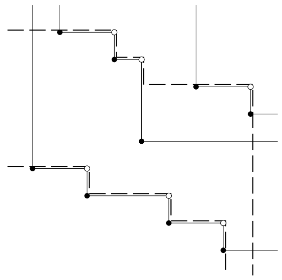
Wir werden hier keinen vollständigen Beweis dieses Satzes geben aber einige der Ideen andeuten.

Sei $\mathcal{A}_X^k$ die Menge der k-Antiketten von X und $\hat{\mathcal{A}}_X^k$ die Menge der k-

Antiketten die keine $(k-1)$-Antiketten von X sind. Elemente von $\hat{\mathcal{A}}_X^k$ werden auch *strikte k-Antiketten* genannt. Wir definieren nun zwei Abbildungen zwischen $\mathcal{A}_X^k$ und $\mathcal{A}_{S(X)}^k$.

(1) $\delta : \mathcal{A}_X^k \to \mathcal{A}_{S(X)}^k$
Sei $\{A_1, .., A_k\}$ die kanonische Antikettenzerlegung einer k-Antikette in X. Wir sammeln nun jene Skelettpunkte die auf Sprunglinien $\partial\sigma(A_i)$ liegen. Für jedes i ist die Menge $\partial\sigma(A_i) \cap S(X)$ eine Antikette im Skelett, also ist $\bigcup_{i=1}^k \partial\sigma(A_i) \cap S(X)$ in $\mathcal{A}_{S(X)}^k$.

(2) $\Delta : \mathcal{A}_{S(X)}^k \to \mathcal{A}_X^k$
Sei $\{A_1, .., A_k\}$ die dual kanonische Antikettenzerlegung einer k-Antikette in $S(X)$. Wir sammeln nun jene Punkte von X die auf dualen Sprunglinien $\partial\sigma_{\swarrow}(A_i)$ liegen. Für jedes i ist die Menge $\partial\sigma_{\swarrow}(A_i) \cap X$ eine Antikette, also ist $\bigcup_{i=1}^k \partial\sigma_{\swarrow}(A_i) \cap X$ in $\mathcal{A}_X^k$. Der Δ-Operator wird im nächsten Bild illustriert.



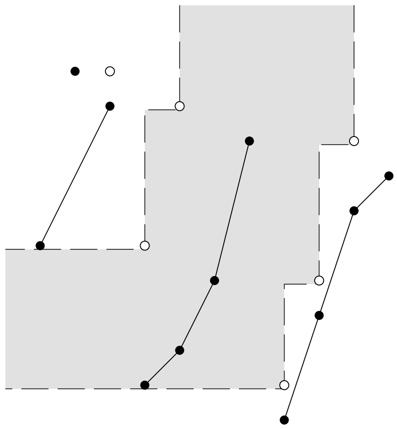
Die Bedeutung der Operatoren Δ und δ beruht auf.

- Für $Y \in \hat{\mathcal{A}}_X^k$ ist $|\delta(Y)| \geq |Y| - k$.
- Für $Y \in \hat{\mathcal{A}}_{S(X)}^k$ ist $|\Delta(Y)| \geq |Y| + k$.
- Ist $Y \in \hat{\mathcal{A}}_{S(X)}^k$ eine maximum k-Antikette in $S(X)$, dann ist $|\Delta(Y)| = |Y| + k$ und $\Delta(Y)$ ist eine maximum k-Antikette in $S(X)$.

Aus diesen Aussagen folgert man die eine Hälfte des Satzes. Implizit ist damit auch ein Algorithmus zur Berechnung von maximum k-Antiketten in 2-dimensionalen Ordnungen gegeben.

Wir wenden uns nun dem Fall der ℓ-Ketten zu. Die *kanonische Kettenzerlegung* einer Punktmenge X, entspreche der kanonische Antikettenzerlegung der an einer vertikalen gespiegelten Menge X^R, oder anders, der duch die Schatten $\sigma_{\nwarrow}$ bezüglich einer von rechts unten kommenden Beleuchtung gegebene

Zerlegung. Sei Y eine ℓ-Kette in $S(X)$ und $\{C_1, .., C_\ell\}$ die kanonische Kettenzerlegung. Für i zwischen 2 und ℓ sei die *i-te Region* $R_i(Y)$ sei definiert als $\sigma_\nwarrow(C_{i-1}) - \sigma_\nwarrow(C_i)$. Die Definition und die gesondert zu definierenden Regionen R_1 und $R_{\ell+1}$ werden im folgenden Bild dargestellt.



Wir beschreiben nun einen Algorithmus zur Berechnung einer maximum ℓ-Kette in X. (Siehe auch voriges Bild)

1. Berechne das Skelett $S(X)$ von X.
2. Berechne rekursiv eine maximum $(\ell - 1)$-Kette in $S(X)$.
3. In jeder der ℓ durch Y gegebenen Regionen berechne eine maximum Kette von $X \cup R_i(Y)$. Die Vereinigung dieser ℓ-Ketten ist eine maximum ℓ-Kette von X.

Die durch dieses Verfahren berechnete ℓ-Kette in X enthält zumindest $|Y| + \lambda$ Punkte, wenn λ die Höhe von X, also die Länge einer längsten Kette in X ist. Ist gezeigt, daß das Verfahren tatsächlich eine maximum ℓ-Kette berechnet, dann folgt der zweite Teil des Satzes durch Induktion.

Zum Kapitel

Das Standardwerk zur Dimension von Ordnungen ist von Trotter [48]. Permutationsgraphen und 2-dimensionale Ordnungen werden außer in [48] auch in [18, 38, 43] behandelt. Der Erkennungsalgorithmus basiert auf [33]. Es sei noch angemerkt, daß die Erkennung von Ordnungen der Dimension $k \geq 3$ nach einem Resultat von Yannakakis NP-vollständig ist.

Der Begriff der k-Perfektheit wurde von Lovász eingeführt. Die k-Perfektheit von Line-Graphen bipartiter Graphen ist von Saks.

Die Originalbeweise der Sätze 8–10 findet man in [19, 20, 21]. Einfachere Beweise wurden von Saks [42], Fomin [14] und Frank [15] gefunden. Der auf dem *minimum cost flow* von Ford und Fulkerson beruhende Beweis von Frank wurde von Felsner [11] noch weiter vereinfacht und auf gerichtete Graphen verallgemeinert. Dieser letztgenannte Beweis impliziert, daß maximum k-Antiketten und ℓ-Ketten sowie die dazu dualen optimale Zerlegungen in $O(n^3)$ Zeit gefunden werden können.

Der klassische Beweis des satzes von Robinson und Shensted wird mit dem sogenannten *bumping Algorithmus* geführt. Dieser Zugang ist etwa in [44] beschrieben. Der Begriff des Skeletts einer Punktemenge stammt von Viennot in dem Übersichtsartikel [51] werden zahlreiche Anwendungen dieses Begriffs gegeben. Der Beweis von Satz 15 ist in [12] ausgearbeitet. Der dort vorgeschlagene Algorithmus zur Berechnung von maximum ℓ-Ketten läuft in $O(\ell n \log n)$ Zeit.

Eine sehr umfassende Übersicht zu den in diesem Kapitel behandelten Fragen und allen Möglichen Verallgemeinerungen und Varianten gibt West [45].

Literatur

[1] M. Aigner, *Graphentheorie, eine Entwicklung aus dem 4-Farben Problem*, Teubner Verlag, Stuttgart, 1984.

[2] C. Berge and V. Chvátal (eds.), *Topics on Perfect Graphs*, Math. stud. 88, North-Holland, Amsterdam-New York-Oxford, 1984.

[3] N.L. Biggs, E.K. Lloyd and R.J. Wilson, *Graph Theory 1736-1936*, Clarendon Press, Oxford, 1976.

[4] H. Bodländer, *A tourists guide to treewidth*, Acta Cybernetica **11** (1993), 1–21.

[5] A. Brandstädt, *Special graph classes – a survey*, Schriften-Mathematik Uni.-Duisburg (1991).

[6] A. Brandstädt, *Graphen und Algorithmen*, Teubner, Stuttgart, 1994.

[7] V. Chvátal, *Star-cutsets and perfect graphs*, J. Combin. Theory Ser. B **39** (1985), 189–199.

[8] V. Chvátal and C. T. Hoàng, *On the P_4-structure of perfect graphs I. Even decompositions*, J. Combin. Theory Ser. B **39** (1985), 209–219.

[9] V. Chvátal, W. J. Lenhart, and N. Sbihi, *Two-colorings that decompose perfect graphs*, J. Combin. Theory Ser. (B) **49** (1990), 1–9.

[10] T. Cormen, C. Leiserson, R. Rivest, *Introduction to Algorithms*, MIT Press, Cambridge, Mass., 1990.

[11] S. Felsner, *Orthogonal Structures in Directed Graphs*, J. Combin. Theory Ser. (B) **57** (1993), 309–321.

[12] S. Felsner and L. Wernisch, *Maximum k-Chains in Planar Point Sets: Combinatorial Structure and Algorithms*, to appear SIAM Journal of Computing 1996.

[13] P.C. Fishburn, *Interval Orders and Interval Graphs*, Wiley, New-York, 1985.

[14] S.V. Fomin, *Finite partially ordered sets and Youg Tableaux*, Soviet Math. Dokl. **19** (1978), 1510–1514.

[15] A. Frank, *On chain and antichain families of a partially ordered set*, J. Combin. Theory Ser. (B) **29** (1980), 176–184.

[16] T. Gallai, *Transitiv orientierbare Graphen*, Acta Math. Acad. Scient. Hung. **18** (1967), 25–66.

[17] M. Garey and D. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.

[18] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[19] C. Greene, *An extension of Schensted's theorem*, Adv. in Math., **14** (1974), 254–265.

[20] C. Greene, *Some partitions associated with a partially ordered set* J. Comb. Th. (A) **20** (1976) 69–79.

[21] C. Greene and D. J. Kleitman, *The structure of sperner k-families*, J. Comb. Th. (A) **20** (1976), 41–68.

[22] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms in Combinatorial Optimization*, Springer-Verlag, Berlin-New York, 1988.

[23] R. B. Hayward, *Weakly triangulated graphs*, J. Combin. Theory Ser. B **39** (1985), 200–208.

[24] R. B. Hayward, C. Hoàng, and F. Maffray, *Optimizing weakly triangulated graphs*, Graphs Combin. **5** (1989), 339–349.

[25] C. T. Hoàng, *On the P_4-structure of perfect graphs II. Odd decompositions*, J. Combin. Theory Ser. B **39** (1985), 220–232.

[26] D. Jungnickel, *Graphen, Netzwerke und Algorithmen*, Bibliographisches Institut, Zürich, 1987.

[27] T. Kloks, *Treewidth, Computations and Approximations*, Lecture Notes in Computer Science 842, Springer 1994.

[28] D. E. Knuth, *The Sandwich Theorem*, Electr. J. Combin. **1** (1994).

[29] C.G. Lekkerkerker and J.C. Boland, *Representation of a finite Graph by a set of intervals on the real line*, Fund. Math. **51** (1962), 45–64.

[30] L. Lovász, *Perfect graphs*, in More Selected Topics on Graph Theory (L. M. Beineke and R. L. Wilson, eds.), Academic Press, London-New York, 1983, pp. 55–87.

[31] L. Lovász, *An Algorithmic Theory of Numbers, Graphs and Convexity*, CBMS-NSF Regional Conf. Series in Appl. Math., number 50, SIAM, Philadelphia, 1986.

[32] L. Lovász, *Stable sets and polynomials*, Discrete Math. **124** (1994), 137–153.

[33] R.M. McConnell and J.P. Spinrad, *Linear-Time Modular Decomposition and Efficient Transitive Orientation of Comparability Graphs*, Fifth ACM-SIAM Symp. on Discr. Algo. (1994), 536-545.

[34] R.M. McConnell and J.P. Spinrad, *Modular Decomposition and Transitive Orientation*, http://www.cs.amherst.edu/∼ross/work.html, 47 pages.

[35] H. Meyniel, *A new property of critical imperfect graphs and some consequences*, European J. Combin. **8** (1987), 313–316.

[36] M. Middendorf and F. Pfeiffer, *On the complexity of recognizing perfectly orderable graphs*, Discrete Math. **80** (1990), 327-333.

[37] R.H. Möhring, *Algorithmic aspects of comparability graphs and interval graphs*, in Rival (ed.), *Graphs and Order*, Reidel Publishing Company, 1985, 41-101.

[38] R.H. Möhring, *Computationally tractable classes of ordered sets*, in Rival (ed.), *Algorithms and Order*, Reidel Publishing Company, 1989, 105-193.

[39] R.H. Möhring and F.J. Radermacher, *Substitution decomposition for discrete structures and connections with combinatorial optimization*, Annals of Discrete Mathematics **19** (Burkhard el al Eds), 1984, 257–356.

[40] H.J. Prömel und A. Steger, *Allmost all Berge graphs are prefekt*, Combin. Prob. and Computing **1** (1992), 53–79.

[41] B. A. Reed, *A note on even pairs*, Discrete Math. **65** (1987), 317–318.

[42] M. Saks, *A short proof for the existence of k-saturated partitions of partially ordered sets*, Advances in math. **33** (1979), 207–211.

[43] K. Simon, *Effiziente Algorithmen für perfekte Graphen*, Teubner, Stuttgart, 1992.

[44] D. Stanton and D. White, *Constructive Combinatorics*, Undergraduate Texts in Math, Springer Verlag, 1986.

[45] D.B. West, *Parameters of Partial Orders and Graphs: Packing, Covering and Representation*, in Rival (ed.), *Graphs and Order*, Reidel Publishing Company, 1985, 267–350.

[46] R.J. Wilson, *Introduction to Graph Theory*(Second Edition), Longman, London, 1979.

[47] R. E. Tarjan and M. Yannakakis, *Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs*, SIAM J. Computing **13** (1984), 566–579.

[48] W.T. Trotter, *Combinatorics and Partially Ordered Sets: Dimension Theory*, Johns Hopkins Press, Baltimore, 1991.

[49] W.T Trotter, Partially Ordered Sets, in R.L. Graham, M. Grötschel and L. Lovász (eds.), *Handbook of Combinatorics, Volume I*, North-Holland, Amsterdam, 1995.

[50] L. Vandenberghe und S. Boyd, *Semidefinite Programming*, SIAM Review (1996).

[51] G. Viennot, *Chain and antichain families, grids and young tableaux*, in M. Pouzet (editor) *Orders: Description and Roles*, pages 409–463. North-Holland Math. Stud. 99, Amsterdam-New York, 1984.