Proof Theory

Helmut Schwichtenberg

Mathematisches Institut der Universität München
Sommersemester 1994

Preface

These notes represent the content of a lecture course on proof theory given during Sommersemester 1994 at the Mathematisches Institut der Universität München. They are still rather sketchy and will have to undergo many more revisions.

For their help in preparing these notes I would like to thank Ulrich Berger, Michael Bopp, Felix Joachimski, Ralph Matthes, Karl–Heinz Niggl, Jaco van de Pol and Robert Stärk.

München, July 1994

Helmut Schwichtenberg

Contents

Part I: Preliminaries
1 Typed languages 1
2 Natural deduction 6
3 Hilbert style systems 13
4 Normalization 15
5 The strong existential quantifier 38
6 Realizing terms 43
7 Arithmetic 47

Part II: Computational content of proofs
8 A–translation with program extraction and the direct method 54
9 The root example; refinements 63

Part III: Classifying arithmetical proofs
10 Ordinals below epsilon zero 73
11 Provability of initial cases of transfinite induction 76
12 Normalization for arithmetic with the omega rule 80
13 Unprovable initial cases of transfinite induction 83
14 Normalization for arithmetic is impossible 86

Appendix
15 Permutative conversions 87
References 94
Index 97

Part I: Preliminaries

1 Typed languages

Let us first fix our language $\mathcal{L}$. Let G be a set of ground types (e.g. nat and boole). *Types* (also called object types or simple types) are formed from G by the operations $\rho \to \sigma$ and $\rho \times \sigma$. The *level* of a type ρ is defined by

$$\begin{aligned}
\mathrm{lev}(\iota) &:= 0 \quad \text{for any ground type } \iota, \\
\mathrm{lev}(\rho \to \sigma) &:= \max(\mathrm{lev}(\rho) + 1, \mathrm{lev}(\sigma)), \\
\mathrm{lev}(\rho \times \sigma) &:= \max(\mathrm{lev}(\rho), \mathrm{lev}(\sigma)).
\end{aligned}$$

For any type ρ let a countable infinite set of *variables* of type ρ be given. We denote variables of type ρ by $x^\rho, y^\rho, \ldots$. We also assume that a set $\mathcal{C}$ of *constants* denoted c^ρ is given, each of an arbitrary type ρ. Furthermore, we assume that a set $\mathcal{F}$ of *function symbols* denoted f is given, each of a "functionality" $(\rho_1, \ldots, \rho_m) \to \sigma$. They are intended to denote external functionals, which need not be represented by objects in the model. Note that it is a consequence of this that we can not freely abstract variables in a term, for otherwise we could form $\lambda x^\rho\, f(x)$ for an arbitrary function symbol f, and hence f would be an object of our model.

We define inductively *terms* t^ρ of type ρ, the set $\mathrm{FV}(t^\rho)$ of variables free in t^ρ and the set $\mathrm{nonabs}(t^\rho)$ of non–abstractable variables in $\mathrm{FV}(t^\rho)$.

- x^ρ and c^ρ are terms of type ρ, $\mathrm{FV}(x^\rho) = \{x^\rho\}$, $\mathrm{FV}(c^\rho) = \emptyset$, $\mathrm{nonabs}(x^\rho) = \emptyset$, $\mathrm{nonabs}(c^\rho) = \emptyset$.
- If $t_1^{\rho_1}, \ldots, t_n^{\rho_n}$ are terms and $f \in \mathcal{F}$ a function symbol of functionality $(\rho_1, \ldots, \rho_m) \to \sigma$, then $f(t_1, \ldots, t_n)$ is a term of type σ. $\mathrm{FV}(f(t_1, \ldots, t_n)) = \mathrm{FV}(t_1) \cup \cdots \cup \mathrm{FV}(t_n)$, $\mathrm{nonabs}(f(t_1, \ldots, t_n)) = \mathrm{FV}(t_1) \cup \cdots \cup \mathrm{FV}(t_n)$
- If t is a term of type σ and $x^\rho \notin \mathrm{nonabs}(t)$, then $\lambda x^\rho\, t$ is a term of type $\rho \to \sigma$. $\mathrm{FV}(\lambda x^\rho\, t) = \mathrm{FV}(t) \setminus \{x^\rho\}$, $\mathrm{nonabs}(\lambda x^\rho\, t) = \mathrm{nonabs}(t)$.
- If t is a term of type $\rho \to \sigma$ and s is a term of type ρ, then ts is a term of type σ. $\mathrm{FV}(ts) = \mathrm{FV}(t) \cup \mathrm{FV}(s)$, $\mathrm{nonabs}(ts) = \mathrm{nonabs}(t) \cup \mathrm{nonabs}(s)$.
- If t_i is a term of type ρ_i for $i \in \{0, 1\}$, then $\langle t_0, t_1 \rangle$ is a term of type $\rho_0 \times \rho_1$. $\mathrm{FV}(\langle t_0, t_1 \rangle) = \mathrm{FV}(t_0) \cup \mathrm{FV}(t_1)$, $\mathrm{nonabs}(\langle t_0, t_1 \rangle) = \mathrm{nonabs}(t_0) \cup \mathrm{nonabs}(t_1)$.
- If t is a term of type $\rho_0 \times \rho_1$ and $i \in \{0, 1\}$, then $\pi_i(t)$ is a term of type ρ_i. $\mathrm{FV}(\pi_i(t)) = \mathrm{FV}(t)$, $\mathrm{nonabs}(\pi_i(t)) = \mathrm{nonabs}(t)$.

We now fix our notion of a *formula*. Let an additional ground type $\circ$ be given, to be viewed as the type of propositions. $\circ$ is not to be used in object types $\rho, \sigma, \cdots$. We assume that a set $\mathcal{P}$ of *relation symbols* R of functionalities $(\rho_1, \dots, \rho_n) \to \circ$ is given. 0–ary relation symbols are called propositional symbols. *Formulas* are defined inductively by

- If $t_1^{\rho_1}, \dots, t_n^{\rho_n}$ are terms and $R \in \mathcal{P}$ a relation symbol of functionality $(\rho_1, \dots, \rho_n) \to \circ$, then $R(t_1, \dots, t_n)$ is a formula.
- $\bot$ (to be read "falsity") is a formula.
- If A and B are formulas, then $A \to B$ is a formula.
- If A and B are formulas, then $A \wedge B$ is a formula.
- If A is a formula and x^ρ is a variable, then $\forall x^\rho\, A$ is a formula.

$R(t_1, \dots, t_n)$ and $\bot$ are called *atomic formulas* or *atoms*. Note that $\bot$ is not to a propositional symbol.

We use (also with indices)

$$\begin{aligned} r, s, t &\quad \text{for terms,} \\ x, y, z &\quad \text{for variables,} \\ a, b, c &\quad \text{for constants,} \\ P, Q, R &\quad \text{for relation symbols,} \\ f, g, h &\quad \text{for function symbols,} \\ A, B, C &\quad \text{for formulas.} \end{aligned}$$

Negation, disjunction and the existential quantifier are defined by

$$\begin{aligned} \neg A &:= A \to \bot, \\ A \vee B &:= \neg A \wedge \neg B \to \bot, \\ \exists x\, A &:= \neg \forall x\, \neg A. \end{aligned}$$

The set $\mathrm{FV}(A)$ of variables free in an $\mathcal{L}$–formula A is defined as usual:

$$\begin{aligned} \mathrm{FV}(R(t_1, \dots, t_n)) &:= \mathrm{vars}(t_1) \cup \ldots \cup \mathrm{vars}(t_n), \\ \mathrm{FV}(\bot) &:= \emptyset. \\ \mathrm{FV}(A \to B) &:= \mathrm{FV}(A) \cup \mathrm{FV}(B). \\ \mathrm{FV}(A \wedge B) &:= \mathrm{FV}(A) \cup \mathrm{FV}(B). \\ \mathrm{FV}(\forall x\, A) &:= \mathrm{FV}(A) \setminus \{x\}. \end{aligned}$$

A term t and a formula A are called *closed*, if $\mathrm{FV}(t) = \emptyset$ or $\mathrm{FV}(A) = \emptyset$, respectively. We write $t[x_1, \dots, x_n]$ or $A[x_1, \dots, x_n]$ to indicate that all variables free in t or A are in the list $x_1, \dots, x_n$.

For simplicity we identify terms and formulas which differ only by the names of bound variables. This makes it possible to define substitution $r[t^\rho/x^\rho]$ and $A[t^\rho/x^\rho]$ of a term t^ρ for a variable x^ρ in a particularly simple fashion.

For a term r, variable x^ρ and a term t^ρ we define the result of substituting t^ρ for x^ρ in r inductively on r:

$$\begin{aligned}
y[t/x] &:= \begin{cases} t, & \text{falls } x = y; \\ y, & \text{sonst.} \end{cases} \\
c[t/x] &:= c. \\
f(t_1, \dots, t_n)[t/x] &:= f(t_1[t/x], \dots, t_n[t/x]), \\
(\lambda y^\rho\, r)[t/x] &:= \lambda y^\rho\, r[t/x], \quad \text{where } y \neq x \text{ and } y \notin \mathrm{FV}(t), \\
(rs)[t/x] &:= r[t/x]s[t/x], \\
\langle t_0, t_1 \rangle[t/x] &:= \langle t_0[t/x], t_1[t/x] \rangle, \\
\pi_i(r)[t/x] &:= \pi_i(r[t/x]),
\end{aligned}$$

and siminarly for a formula A, variable x^ρ and a term t^ρ we define the result of substituting t^ρ for x^ρ in A inductively on A:

$$\begin{aligned}
R(t_1, \dots, t_n)[t/x] &:= R(t_1[t/x], \dots, t_n[t/x]), \\
\bot[t/x] &:= \bot, \\
(A \to B)[t/x] &:= A[t/x] \to B[t/x]. \\
(A \wedge B)[t/x] &:= A[t/x] \wedge B[t/x]. \\
(\forall y\, A)[t/x] &:= \forall y\, A[t/x], \quad \text{where } y \neq x \text{ and } y \notin \mathrm{FV}(t).
\end{aligned}$$

We now want to formulate a general notion of a structure or model for our typed languages, called environment model in [3].

$\mathcal{M} = ((D_\rho)_\rho, I_0, I_1)$ is called an *$\mathcal{L}$–structure*, if D_ρ is a nonempty set for any type ρ, I_0 is a mapping assigning to any constant symbol $c^\rho \in \mathcal{C}$ an object $I_0(c) \in D_\rho$ and to any function symbol $f \in \mathcal{F}$ of functionality $(\rho_1, \dots, \rho_n) \to \sigma$ a function

$$I_0(f)\colon D_{\rho_1} \times \cdots \times D_{\rho_n} \to D_\sigma,$$

and I_1 is a mapping assigning to any relation symbol $R \in \mathcal{P}$ of functionality $(\rho_1, \dots, \rho_n) \to \circ$ a relation

$$I_1(R) \subseteq D_{\rho_1} \times \cdots \times D_{\rho_n}.$$

In case $n = 0$ we have $I_1(R) \in \{\text{true}, \text{false}\}$. An *$\mathcal{L}$–environment model* is given by an $\mathcal{L}$–structure $\mathcal{M} = ((D_\rho)_\rho, I_0, I_1)$ and in addition

(i) bijections $\Phi_{\rho,\sigma}\colon D_{\rho\to\sigma} \to [D_\rho \to D_\sigma]$ between $D_{\rho\to\sigma}$ and some set $[D_\rho \to D_\sigma]$ of functions from D_ρ to D_σ,

(ii) bijections $\Psi_{\rho,\sigma}\colon D_{\rho\times\sigma} \to D_\rho \times D_\sigma$, and

(iii) a mapping assigning to any term t and any environment U (i.e. mapping from the set of variables into $\bigcup_\rho D_\rho$ such that for each x^ρ we have $U(x^\rho) \in D_\rho$) an element $[\![t^\rho]\!]_U \in D_\rho$ such that the following holds.

$$
\begin{aligned}
[\![x^\rho]\!]_U &= U(x^\rho), \\
[\![c^\rho]\!]_U &= I_0(c^\rho), \\
[\![f(t_1,\dots,t_n)]\!]_U &= I_0(f)([\![t_1]\!]_U,\dots,[\![t_n]\!]_U), \\
[\![\lambda x^\rho\, t]\!]_U &= \Phi^{-1}_{\rho,\sigma}(f), \quad \text{where } f(a) = [\![t]\!]_{U[a/x^\rho]} \text{ for all } a \in D_\rho, \\
[\![ts]\!]_U &= (\Phi_{\rho,\sigma}([\![t]\!]_U))([\![s]\!]_U), \\
[\![\langle t_0, t_1\rangle]\!]_U &= \Psi^{-1}_{\rho,\sigma}\left([[\![t_0]\!]_U, [\![t_1]\!]_U]\right), \\
[\![\pi_i(t)]\!]_U &= (\Psi_{\rho,\sigma}([\![t]\!]_U))_i \quad \text{for } i \in \{0,1\},
\end{aligned}
$$

Note that for a given $\mathcal{L}$–structure $\mathcal{M} = ((D_\rho)_\rho, I_0, I_1)$ and bijections $\Phi_{\rho,\sigma}$ and $\Psi_{\rho,\sigma}$ it is in general not possible to *define* $[\![t]\!]_U$ by the requirements above. The reason is that the function $f\colon D_\rho \to D_\sigma$ defined by $f(a) = [\![t]\!]_{U[a/x^\rho]}$ may not be in the range $[D_\rho \to D_\sigma]$ of $\Phi_{\rho,\sigma}$, and hence we cannot define $[\![\lambda x^\rho\, t]\!]_U$ by $\Phi^{-1}_{\rho,\sigma}(f)$. — A trivial way out of this difficulty is of course to let $[D_\rho \to D_\sigma]$ be the set of *all* functions from D_ρ to D_σ. Hence we obtain the following trivial

Example of an environment model. Let D_ι for any ground type ι be an arbitrary nonempty set. Choose $D_{\rho\to\sigma}$ to be the set of *all* functions from D_ρ to D_σ, and $D_{\rho\times\sigma} = D_\rho \times D_\sigma$. If we then take $\Phi_{\rho,\sigma}$ and $\Psi_{\rho,\sigma}$ to be identities, we can define $[\![t]\!]_U$ by the equations above.

There are more interesting examples of environment models, most notably the model of the partial continuous function due to Scott [27] and Ersov [9]; see also [24] for an exposition.

We will usually be somewhat sloppy with our notation and leave out Φ and Ψ. So we write

$$
\begin{aligned}
[\![t]\!]_U[\![s]\!]_U &\quad \text{instead of} \quad (\Phi_{\rho,\sigma}([\![t]\!]_U))([\![s]\!]_U), \\
([\![t]\!]_U)_i &\quad \text{instead of} \quad (\Psi_{\rho,\sigma}([\![t]\!]_U))_i, \\
[[\![t_0]\!]_U, [\![t_1]\!]_U] &\quad \text{instead of} \quad \Psi^{-1}_{\rho,\sigma}\left([[\![t_0]\!]_U, [\![t_1]\!]_U]\right).
\end{aligned}
$$

We often write $f^{\mathcal{M}}$ for the interpretation $I_0(f)$ of a function symbol and $R^{\mathcal{M}}$ for the interpretation $I_1(R)$ of a relation symbol.

For any environment model $\mathcal{M}$, environment U and formula A we define a model relation $\mathcal{M} \models A[U]$ by induction on A.

$$
\begin{aligned}
\mathcal{M} \models (R(t_1,\dots,t_n))[U] &\iff [\![t_1]\!]_U,\dots,[\![t_n]\!]_U \in R^{\mathcal{M}}. \\
\mathcal{M} \not\models \bot[U] & \\
\mathcal{M} \models (A \to B)[U] &\iff \text{if } \mathcal{M} \models A[U], \text{ then } \mathcal{M} \models B[U]. \\
\mathcal{M} \models (A \land B)[U] &\iff \mathcal{M} \models A[U] \text{ and } \mathcal{M} \models B[U]. \\
\mathcal{M} \models (\forall x\, A)[U] &\iff \text{for all } a \in |\mathcal{M}| \text{ we have } \mathcal{M} \models A[U[a/x]].
\end{aligned}
$$

Coincidence Lemma. *(i) If* $U(x) = U'(x)$ *for all* $x \in \mathrm{FV}(t)$*, then* $[\![t]\!]_U = [\![t]\!]_{U'}$.

(ii) If $U(x) = U'(x)$ *for all* $x \in \mathrm{FV}(A)$*, then* $\mathcal{M} \models A[U] \iff \mathcal{M} \models A[U']$.

Proof by induction on t and A. □

Substitution Lemma. *(i)* $[\![r[t/x]]\!]_U = [\![r]\!]_{U'}$ *with* $U' = U[[\![t]\!]_U/x]$.

(ii) $\mathcal{M} \models (A[t/x])[U] \iff \mathcal{M} \models A[U']$ *with* $U' = U[[\![t]\!]_U/x]$.

Proof by induction on r and A, using the Coincidence Lemma. □

2 Natural deduction

As our deductive formalism we use the system of natural deduction introduced by Gerhard Gentzen in [11]. It consists of the following introduction and elimination rules for $\to$, $\wedge$ and $\forall$.

For any $\mathcal{L}$–formula A let countably many *assumption variables* of type A be given. We use u^A, v^A, w^A to denote assumption variables of type A.

Later we will define substitution $d[t/x]$ of object terms t for object variables x in derivation terms d. We want to avoid that different assumption variables are identified by a substitution of object terms, as e.g. in $\langle u_0^{R(t)}, u_0^{R(x)}\rangle[t/x]$. Therefore we assume that for any two assumption variables u_i^A, u_j^B in a derivation d in case $A \neq B$ we also have $i \neq j$ (cf. [33], p. 30).

The notions of a *derivation term* d^A in minimal logic and its set $\mathrm{FA}(d^A)$ of *free assumption variables* are defined inductively by

(A) u^A is a derivation term with $\mathrm{FA}(u^A) = \{u^A\}$.

($\to^+$) If d^B is a derivation term, then

$$(\lambda u^A\, d^B)^{A\to B}$$

is a derivation term with $\mathrm{FA}(\lambda u^A\, d^B) = \mathrm{FA}(d^B) \setminus \{u^A\}$.

($\to^-$) If $d^{A\to B}$ and e^A are derivation terms, then

$$(d^{A\to B} e^A)^B$$

is a derivation term with $\mathrm{FA}(d^{A\to B} e^A) = \mathrm{FA}(d^{A\to B}) \cup \mathrm{FA}(e^A)$.

($\wedge^+$) If d^B and e^B are derivation terms, then

$$\langle d^A, e^B\rangle^{A\wedge B}$$

is a derivation term with $\mathrm{FA}(\langle d^A, e^B\rangle^{A\wedge B}) := \mathrm{FA}(d^A) \cup \mathrm{FA}(e^B)$.

($\wedge^-$) If $d^{A_0\wedge A_1}$ is a derivation term, then

$$\pi_i(d^{A_0\wedge A_1})^{A_i}$$

is a derivation term with $\mathrm{FA}(\pi_i(d^{A_0\wedge A_1})^{A_i}) := \mathrm{FA}(d^{A_0\wedge A_1})$ (for $i \in \{0,1\}$).

($\forall^+$) If d^A is a derivation term and $x^\rho \notin \bigcup\{\mathrm{FV}(B) | u^B \in \mathrm{FA}(d^A)\}$, then

$$(\lambda x^\rho\, d^A)^{\forall x^\rho\, A}$$

is a derivation term with $\mathrm{FA}(\lambda x^\rho\, d^A) = \mathrm{FA}(d^A)$.

($\forall^-$) If $d^{\forall x^\rho\, A}$ is a derivation term and t^ρ is a term, then

$$(d^{\forall x^\rho\, A} t^\rho)^{A[t/x^\rho]}$$

is a derivation term with $\mathrm{FA}(d^{\forall x^\rho\, A} t^\rho) = \mathrm{FA}(d^{\forall x^\rho\, A})$.

It is sometimes useful to display derivation terms in the following graphical fashion.

($\to^+$)

$$\dfrac{\begin{matrix} u\colon A \\ | \\ B \end{matrix}}{A \to B}\to^+ u$$

($\to^-$)

$$\dfrac{\begin{matrix} | \\ A \to B \end{matrix} \qquad \begin{matrix} | \\ A \end{matrix}}{B}\to^-$$

($\wedge^+$)

$$\dfrac{\begin{matrix} | \\ A_0 \end{matrix} \qquad \begin{matrix} | \\ A_1 \end{matrix}}{A_0 \wedge A_1}\wedge^+$$

($\wedge_i^-$)

$$\dfrac{\begin{matrix} | \\ A_0 \wedge A_1 \end{matrix}}{A_i}\wedge_i^-$$

($\forall^+$)

$$\dfrac{\begin{matrix} | \\ A \end{matrix}}{\forall x\, A}\forall^+ x$$

provided x does not occur free in any open assumption of the given derivation of A.

($\forall^-$)

$$\dfrac{\begin{matrix} | \\ \forall x\, A \end{matrix} \qquad t}{A[t/x]}\forall^-$$

A derivation term d^A is called *closed*, if $\mathrm{FA}(d^A) = \emptyset$. We write

$$d^B[u_1^{A_1}, \dots, u_n^{A_n}]$$

to indicate that the assumption variables free in d^B are in the list $u_1^{A_1}, \dots, u_n^{A_n}$. We also use the notation $d\colon A$ instead of d^A.

Definition. *A formula A is called derivable from assumptions $A_1, \dots, A_n$, is there is a derivation term $d^B[u_1^{A_1}, \dots, u_n^{A_n}]$ with different assumption variables $u_1^{A_1}, \dots, u_n^{A_n}$.*

Let S be a (finite or infinite) set of formulas. We write $S \vdash B$, if the formula B is derivable from finitely many assumptions $A_1, \dots, A_n \in S$.

Examples for derivable formulas are

$$\begin{aligned}
&P \to (Q \to P),\\
&(P \to Q \to R) \to (P \to Q) \to P \to R,\\
&(\forall x. P(x) \to Q(x)) \to \forall x\, P(x) \to \forall x\, Q(x).
\end{aligned}$$

Definition. *For any derivation d we define its set $\mathrm{FV}(d)$ of free (object) variables by*

$$\begin{aligned}
\mathrm{FV}(u^A) &:= \mathrm{FV}(A).\\
\mathrm{FV}(\lambda u^A d^B) &:= \mathrm{FV}(A) \cup \mathrm{FV}(d^B).\\
\mathrm{FV}(d^{A\to B} e^A) &:= \mathrm{FV}(d^{A\to B}) \cup \mathrm{FV}(e^A).\\
\mathrm{FV}(\langle d^A, e^B\rangle) &:= \mathrm{FV}(d^A) \cup \mathrm{FV}(e^B).\\
\mathrm{FV}(\pi_i(d^{A\wedge B})) &:= \mathrm{FV}(d^{A\wedge B}).\\
\mathrm{FV}(\lambda x\, d^A) &:= \mathrm{FV}(d^A) \setminus \{x\}.\\
\mathrm{FV}(d^{\forall x\, A} t) &:= \mathrm{FV}(d^{\forall x\, A}) \cup \mathrm{FV}(t).
\end{aligned}$$

Example: Let d be the derivation given by

$$\frac{u\colon R(x)}{R(x) \to\ R(x)} \to^+ u$$

Then we have

$$\begin{aligned}
d &= \lambda u^{R(x)}\, u^{R(x)},\\
\mathrm{FA}(d) &= \emptyset,\\
\mathrm{FV}(d) &= \{x\}.
\end{aligned}$$

For derivation terms we have two kinds of substitution: we can substitute a derivation term f^A for an assumption variable u^A, and we can substitute an object term t for an object variable x. These substitutions are defined as follows. For simplicity we again identify derivation terms which differ only in the names of bound variables.

Definition.

$$
\begin{aligned}
v[f/u] &:= \begin{cases} f, & \textit{if } u = v; \\ v, & \textit{otherwise.} \end{cases} \\
(\lambda v\, d)[f/u] &:= \lambda v\, d[f/u], \quad \textit{where } u \neq v \textit{ and } v \notin \mathrm{FA}(f). \\
(de)[f/u] &:= d[f/u]e[f/u]. \\
(\langle d, e\rangle)[f/u] &:= \langle d[f/u], e[f/u]\rangle. \\
(\pi_i(d))[f/u] &:= \pi_i(d[f/u]). \\
(\lambda x\, d)[f/u] &:= \lambda x\, d[f/u], \quad \textit{where } x \notin \bigcup\{\mathrm{FV}(A) : v^A \in \mathrm{FA}(f)\}. \\
(dt)[f/u] &:= d[f/u]t.
\end{aligned}
$$

Definition.

$$
\begin{aligned}
u^A[t/x] &:= u^{A[t/x]}. \\
(\lambda u^A d)[t/x] &:= \lambda u^{A[t/x]} d[t/x]. \\
(de)[t/x] &:= d[t/x]e[t/x]. \\
(\langle d, e\rangle)[t/x] &:= \langle d[t/x], e[t/x]\rangle. \\
(\pi_i(d))[t/x] &:= \pi_i(d[t/x]). \\
(\lambda y\, d)[t/x] &:= \lambda y\, d[t/x], \quad \textit{where } x \neq y \textit{ and } y \notin \mathrm{FV}(t). \\
(dr)[t/x] &:= d[t/x]r[t/x].
\end{aligned}
$$

Recall here the requirement for derivation terms mentioned above: for any two assumption variables u_i^A, u_j^B in a derivation d in case $A \neq B$ we also have $i \neq j$ (cf. [33], p. 30).

Lemma. *(i) If d, f are derivation terms and t is an object term, then $d[f/u]$ and $d[t/x]$ are derivation terms.*

(ii) $\mathrm{FA}(d[f/u]) \subseteq (\mathrm{FA}(d) \setminus \{u\}) \cup \mathrm{FA}(f)$.

(iii) $\mathrm{FV}(d[f/u]) \subseteq \mathrm{FV}(d) \cup \mathrm{FV}(f)$.

(iv) $\mathrm{FA}(d[t/x]) = \{u^{A[t/x]} : u^A \in \mathrm{FA}(d)\}$.

(v) $\mathrm{FV}(d[t/x]) \subseteq (\mathrm{FV}(d) \setminus \{x\}) \cup \mathrm{FV}(t)$.

Proof by simultaneous induction (cf. [33], p. 30/31). □

Derivation terms in *intuitionistic* and in *classical* logic are obtained by adding to the first (assumption–) clause of the definition

- in the case of intuitionistic logic: For any relation symbol $R \in \mathcal{P}$ we let

$$\mathrm{Efq}_R \colon \forall \vec{x}. \bot \to R(\vec{x})$$

be a derivation term with $\mathrm{FA}(\mathrm{Efq}_R) = \emptyset$ (Ex–falso–quodlibet axiom).

- in the case of classical logic: For any relation symbol $R \in \mathcal{P}$ we let

$$\mathrm{Stab}_R \colon \forall \vec{x}. \neg\neg R(\vec{x}) \to R(\vec{x})$$

be a derivation term with $\mathrm{FA}(\mathrm{Stab}_R) = \emptyset$ (Stability axiom).

Clearly $\mathrm{FV}(\mathrm{Stab}_R) := \mathrm{FV}(\mathrm{Efq}_R) := \emptyset$.

We write $\Gamma \vdash A$ ($\Gamma \vdash_I A$, $\Gamma \vdash_C A$), if there is a derivation term d^A in minimal (intuitionistic, classical) logic such that for any $u^B \in \mathrm{FA}(d)$ we have $B \in \Gamma$.

Here are some more interesting examples: The Peirce formula

$$((P \to Q) \to P) \to P$$

is derivable in classical, but not in minimal logic. The Mints formula

$$((((P \to Q) \to P) \to P) \to Q) \to Q$$

is derivable in minimal logic. However, its variant

$$((((P \to Q) \to P) \to P) \to R) \to R$$

is derivable in classical but not in minimal logic.

By obvious reasons the stability axiom is also called the *principle of indirect proof* for the relation symbol R. We now want to show that from our stability axioms we can derive the principle of indirect proof for arbitrary formulas (in our $\to \wedge\forall$–language).

Stability Lemma. *From stability assumptions* Stab_R *for any relation symbol* R *occurring in a formula* A *we can derive* $\neg\neg A \to A$.

Proof by induction on A.

Case $R(\vec{t})$. Use Stab_R.

Case $\bot$. $\neg\neg\bot \to \bot \equiv ((\bot \to \bot) \to \bot) \to \bot$

$$\frac{(\bot \to \bot) \to \bot \qquad \bot \to \bot}{\bot}$$

Case $A \to B$. Use

$$\vdash (\neg\neg B \to B) \to \neg\neg(A \to B) \to A \to B.$$

$$\dfrac{\neg\neg B \to B \qquad \dfrac{\dfrac{\neg\neg(A \to B) \qquad \dfrac{\dfrac{\neg B \qquad \dfrac{A \to B \quad A}{B}}{\bot}}{\neg(A \to B)}}{\bot}}{\neg\neg B}}{B}$$

Derivation of $(\neg\neg B \to B) \to \neg\neg(A \to B) \to A \to B$.

$$\dfrac{\neg\neg A \to A \qquad \dfrac{\dfrac{\neg\neg\forall x\, A \qquad \dfrac{\dfrac{\neg A \qquad \dfrac{\forall x\, A \quad x}{A}}{\bot}}{\neg\forall x\, A}}{\bot}}{\neg\neg A}}{A}$$

Derivation of $(\neg\neg A \to A) \to \neg\neg\forall x\, A \to A$.

Case $A \wedge B$. Use

$$\vdash (\neg\neg A \to A) \to (\neg\neg B \to B) \to \neg\neg(A \wedge B) \to A \wedge B.$$

Case $\forall x\, A$. Use

$$\vdash (\neg\neg A \to A) \to \neg\neg\forall x\, A \to A. \qquad \square$$

Similarly we can show that from our ex–falso–quodlibet axioms we can derive ex–falso–quodlibet for arbitrary formulas (again in our $\to \wedge\forall$–language).

Ex–falso–quodlibet Lemma. *From assumptions* Efq_R *for any relation symbol* R *occurring in a formula* A *we can derive* $\bot \to A$ *in intuitionistic logic.*

Proof by induction on A. □

From $\neg\neg A \to A$ one can clearly derive $\bot \to A$. Hence any formula derivable in intuitionistic logic is also derivable in classical logic.

3 Hilbert style systems

By a Hilbert style system we mean a derivation system without the possibility to discharge open assumptions, i.e. without the rule $\to^+$ of $\to$–introduction. Such a system for minimal quantifier logic is the following. Axioms:

$$\begin{aligned}
K&: B \to A \to B,\\
S&: (A \to B \to C) \to (A \to B) \to A \to C,\\
\wedge^+\text{–Ax}&: A \to B \to A \wedge B,\\
\wedge_i^-\text{–Ax}&: A_0 \wedge A_1 \to A_i,\\
\forall^+\text{–Ax}&: (\forall x.A \to B) \to A \to \forall x\, B \qquad \text{if } x \notin \mathrm{FV}(A),\\
\forall^-\text{–Ax}&: \forall x\, A \to A[t/x].
\end{aligned}$$

Rules: only $\to^-$, $\forall^+$. Sometimes $\to^-$ is also called modus ponens, and $\forall^+$ the rule of generalization. Note that in addition to the axioms (or better axiom constants) we also allow assumption variables; the variable condition in the rule $\forall^+$ only refers to the free assumption variables, not to the axiom constants.

We want to prove that this Hilbert system is equivalent to the natural deduction system for minimal quantifier logic introduced in §2.

For the non–trivial direction we consider an intermediate system which has the axioms above and as rules only $\to^+$, $\to^-$, $\forall^+$.

We first show that for any derivation $d[\vec{u}: \vec{A}]: B$ in the natural deduction system we can find a derivation $d'[\vec{u}: \vec{A}]: B$ in the intermediate system. The proof is by induction on d.

Case $\langle d_0, d_1\rangle$. Take $\wedge^+$–Ax $d_0' d_1'$.

Case $\pi_i(d)$. Take $\wedge^-$–Ax d'.

Case dt. Take $\forall^-$–Ax t. □

We now show the so-called deduction theorem, i.e. that for any derivation

$$d[u^A, \vec{u}: \vec{A}]: B$$

in the Hilbert system we can find a derivation $(\lambda^* u^A\, d)[\vec{u}: \vec{A}]: B$ again in the Hilbert system. The proof is again by induction on d.

Case $v \neq u$ or c. Take Kd

Case u. Take SKK

Case de. Take $\lambda^* u^A.de := S(\lambda^* u^A\, d)(\lambda^* u^A\, e)$.

Case $\lambda x\, d$. Take $\lambda^* u^A(\lambda x\, d) := \forall^-\text{–Ax}(\lambda x(\lambda^* u^A\, d))$. □

Now we can easily prove that for any derivation $d[\vec{u}\colon \vec{A}]\colon B$ in the natural deduction system we can find a derivation of B from $\vec{A}$ in the Hilbert system.

For the proof we can assume that d is a derivation in the intermediate system. We now use again induction on d. The only intersting case occurs when the derivation is of the form $\lambda u^A\, d[u^A, \vec{u}\colon \vec{A}]$. By IH we can assume that d already is in the Hilbert system. But then the deduction theorem gives us a derivation $(\lambda^* u^A\, d)[\vec{u}\colon \vec{A}]\colon B$ of B from $\vec{A}$ in the Hilbert system.

Problem: Consider the following variant of the Hilbert system above. Axioms:

$$
\begin{aligned}
K &\colon \forall(B \to A \to B),\\
S &\colon \forall((A \to B \to C) \to (A \to B) \to A \to C),\\
\wedge^+\text{–Ax} &\colon \forall(A \to B \to A \wedge B),\\
\wedge_i^-\text{–Ax} &\colon \forall(A_0 \wedge A_1 \to A_i),\\
K^\forall &\colon \forall(B \to \forall x\, B) \qquad \text{if } x \notin \mathrm{FV}(B),\\
S^\forall &\colon \forall((\forall x.B \to C) \to \forall x\, B \to \forall x\, C),\\
\forall^-\text{–Ax} &\colon \forall(\forall x\, A \to A[t/x]),
\end{aligned}
$$

where $\forall(A)$ denotes the universal closure of the formula A. Rules: only $\to^-$. Again in addition to the axioms we also allow assumption variables. Show that for any derivation $d[\vec{u}\colon \vec{A}]\colon B$ in the natural deduction system we can find a derivation $d'[\vec{u}\colon \vec{A}]\colon B$ in this variant of the Hilbert system.

4 Normalization

We show in this section that any derivation d can be transformed by appropriate conversion steps into a normal form. A derivation in normal form has the property that it does not make “detours”, or more precisely, that it cannot occur that an elimination rule immediately follows an introduction rule. Derivations in normal form have many pleasant properties, and can be used for a variety of results. We also construct in this section the so–called long normal form, by means of an additional conversion step called η–expansion.

Implementing normalization of λ–terms with free variables (like derivations d) in the usual recursive fashion is quite inefficient. However, it is possible to compute the long normal form of such a term by evaluating it in an appropriate model (cf. [2] and also [5]). This makes it possible to use the built–in evaluation mechanism of e.g. Scheme (a Lisp dialect, defined in [4]) to efficiently implement normalization.

We finally show that the requirement to give a normal derivation of a derivable formula can sometimes be unrealistic. Following Orevkov [16] we give examples of formulas C_k which are easily derivable with non–normal derivations (whose number of nodes is linear in k), but which require a non–elementary (in k) number of nodes in any normal derivation.

4.1 Beta–eta normal forms

For the arguments in this section it is convenient to use the following notation.

- By a term we mean a derivation term as well as an object term. $M, N, K, \ldots$ denote terms in this sense. For simplicity we assume that object terms do not contain function symbols.
- $x, y, z \ldots$ denote assumption variables as well as object variables.
- We identify terms differing only by the names of their bound variables.
- $\rho, \sigma, \tau \ldots$ denote formulas as well as types, and ι denotes atomic formulas or $\perp$ as well as ground types. $\rho \times \sigma$ denotes product types as well as conjunctions, and $\rho \to \sigma$ denotes function types as well as implications or universal formulas. $\to$ associates to the right and $\times$ to the left. Furthermore $\times$ has a higher precedence than $\to$.

We now define a *conversion relation* $M \to_0 M'$ between terms. Since we have allowed pairing and components, we have β– and η–conversion for $\to$ as well as for $\times$.

Definition. *$M \to_0 M'$ is defined by*

$$(\lambda x\, M)N \to_0 M[N/x]. \tag{β}$$

$$\lambda x.Mx \to_0 M, \quad \textit{if } x \notin \mathrm{FV}(M) \cup \mathrm{FA}(M). \tag{η}$$

$$\pi_i\langle M_0, M_1\rangle \to_0 M_i \quad \textit{for } i \in \{0,1\}. \tag{$\beta_{\times,i}$}$$

$$\langle \pi_0(M), \pi_1(M)\rangle \to_0 M. \tag{$\eta_\times$}$$

A term M is called β–convertible (η–convertible), if it has the form of a left hand side of (β) or ($\beta_{\times,i}$) ((η) or ($\eta_\times$)). Such terms are also called β–redex or η–redex (for reducible expression).

From $\to_0$ (also denoted by $\to_{0\beta\eta}$) one derives a one–step reduction relation $\to$ (also denoted by $\to_{\beta\eta}$) as follows. Intuitively $M \to M'$ means that M' is obtained from M by converting exactly one subterm.

Definition. *$M \to M'$ is defined inductively by*

$$M \to_0 M' \implies M \to M'.$$

$$M \to M' \implies \lambda x\, M \to \lambda x\, M'. \tag{$+$}$$

$$M \to M' \implies MN \to M'N. \tag{$-_0$}$$

$$N \to N' \implies MN \to MN'. \tag{$-_1$}$$

$$M \to M' \implies \langle M, N\rangle \to \langle M', N\rangle. \tag{$+_{\times,0}$}$$

$$N \to N' \implies \langle M, N\rangle \to \langle M, N'\rangle. \tag{$+_{\times,1}$}$$

$$M \to M' \implies \pi_i(M) \to \pi_i(M') \quad \textit{for } i = 0, 1. \tag{$-_{\times,i}$}$$

Definition. *A term M is $\beta\eta$–normal if M has no β– or η–convertible subterm.*

Hence a term M is $\beta\eta$–normal if and only if M contains no subterm M' such that $M \to M'$. We now show that $\to$ is terminating, i.e. that any reduction sequence starting with M terminates after finitely many steps. By a *reduction sequence* we mean a (finite or infinite) sequence

$$M_1, M_2, \ldots, M_n, \ldots$$

such that M_{i+1} arises from M_i by a β– or η–conversion of a subterm, i.e. $M_i \to M_{i+1}$. We write $M \to^* M'$ (or $M \to^+ M'$) if M' is a member of a reduction sequence (a reduction sequence with at least two elements) starting with M. Hence $\to^*$ is the reflexive transitive closure of $\to$.

For many of our later arguments it is useful to write $M0$ for $\pi_0(M)$ and $M1$ for $\pi_1(M)$.

To prove termination of $\to$ we make use of a method due to W.W. Tait and define so–called *strong computability predicates*. They are defined by induction on the type ρ of a term M^ρ as follows.

- M^ι is strongly computable if M^ι strongly normalizable, i.e. if any reduction sequence starting with M^ι terminates after finitely many steps.

- $M^{\rho\to\sigma}$ is strongly computable if for any strongly computable N^ρ also $(MN)^\sigma$ is strongly computable
- $M^{\rho\times\sigma}$ is strongly computable if $\pi_0(M)^\rho$ and $\pi_1(M)^\sigma$ are strongly computable.

A term M is *strongly computable under substitution* if for any list $\vec{N}$ of strongly computable terms $M[\vec{N}]$ is strongly computable.

Lemma 1. *(i) Any strongly computable term M^ρ is strongly normalizable.*

(ii) Any variable x^ρ is strongly computable.

Proof by simultaneous induction of the type ρ. For ground types ι both claims are trivial.

Case $\rho \to \sigma$. (i) Let $M^{\rho\to\sigma}$ be strongly computable. By IH(ii) and the definition of strong computability $(Mx)^\sigma$ is strongly computable. Hence by IH(i) any reduction sequence starting with Mx terminates. Then clearly this also holds for M, since any subterm of a strongly normalizable term is also strongly normalizable. (ii) Let $\vec{M}$ be a list of strongly computable terms or 0 or 1 such that $x\vec{M}$ is of ground type. We have to show that $x\vec{M}$ is strongly computable, which for a ground type means the same as strongly normalizable. But this follows from IH(i), which says that any reduction sequence starting with some M_i terminates.

Case $\rho\times\sigma$. (i) Let $M^{\rho\times\sigma}$ be strongly computable. By definition e.g. $\pi_0(M)^\rho$ is strongly computable and hence by IH(i) also strongly normalizable. This clearly also holds for M, since any subterm of a strongly normalizable term is also strongly normalizable. (ii) As in Case $\rho \to \sigma$. □

Lemma 2. *If $M \to^* M'$ and M is strongly computable, then so is M'.*

Proof. Let $\vec{N}$ be a list of strongly computable terms or 0 or 1 such that $M\vec{N}$ is of ground type. We have to show that $M'\vec{N}$ is strongly computable, i.e. that any reduction sequence starting with $M'\vec{N}$ terminates. But this clearly holds, since any such sequence can be extended to a reduction sequence starting with $M\vec{N}$. □

Lemma 3. *Any term M is strongly computable under substitution.*

Proof by induction on M.

Case x. The claim follows from Lemma 1(ii).

Case MN. Let $\vec{K}$ be strongly computable. We have to show that $M[\vec{K}]N[\vec{K}]$ is strongly computable. But this clearly holds, since by IH $M[\vec{K}]$ as well as $N[\vec{K}]$ are strongly computable.

Case $\pi_0(M)$, $\pi_1(M)$. Similar.

Case $\lambda x\, M$. Let $\vec{K}$ be strongly computable. We have to show that $\lambda x\, M[\vec{K}]$ is strongly computable. So let N be strongly computable and $\vec{L}$ a list of strongly computable terms or

0 or 1. We have to show that $(\lambda x\, M[\vec{K}])N\vec{L}$ is strongly computable, i.e. that any reduction sequence starting with it terminates. So assume that we have such a reduction sequence. $M[\vec{K}]$, N and $\vec{L}$ are all strongly normalizable. Hence we can assume that in this reduction sequence there is a term $(\lambda x\, M[\vec{K}]')N'\vec{L}'$ with $M[\vec{K}] \to^* M[\vec{K}]'$, $N \to^* N'$ and $\vec{L} \to^* \vec{L}'$ to which in the next step a head conversion has been applied. This must be either a β– or an η–conversion. In both cases the result is

$$M[\vec{K}]'[N']\vec{L}'.$$

Because of $M[\vec{K}] \to^* M[\vec{K}]'$ we also have $M[N, \vec{K}] \to^* M[\vec{K}]'[N']$. Since by IH M is strongly computable under substitution we know that $M[N, \vec{K}]$ is strongly computable, and hence by Lemma 2 also $M[\vec{K}]'[N']$. Since again by Lemma 2 $\vec{L}'$ is strongly computable we obtain that $M[\vec{K}]'[N']\vec{L}'$ is strongly computable and hence strongly normalizable.

Case $\langle M, N\rangle$. Similarly. Let $\vec{K}$ be strongly computable. We have to show that $\langle M[\vec{K}], N[\vec{K}]\rangle$ is strongly computable. So let $\vec{L}$ be a list of strongly computable terms or 0 or 1. We have to show that $\langle M[\vec{K}], N[\vec{K}]\rangle\vec{L}$ is strongly computable, i.e. that any reduction sequence starting with it terminates. So assume that we have such a reduction sequence. $M[\vec{K}]$, $N[\vec{K}]$ and $\vec{L}$ are all strongly normalizable. Hence we can assume that in this reduction sequence there is a term $\langle M[\vec{K}]', N[\vec{K}]'\rangle\vec{L}'$ with $M[\vec{K}] \to^* M[\vec{K}]'$, $N[\vec{K}] \to^* N[\vec{K}]'$ and $\vec{L} \to^* \vec{L}'$ to which in the next step a head conversion has been applied. This must be either a $\beta_{\times,i}$– or an $\eta_\times$–conversion. In case it is a $\beta_{\times,i}$–conversion $\vec{L}'$ must begin with i, hence $\vec{L}' = i, \vec{M}'$ and $\vec{L} = i, \vec{M}$ with $\vec{M} \to^* \vec{M}'$, and e.g. for $i = 0$ the head conversion yields

$$M[\vec{K}]'\vec{M}'.$$

But $M[\vec{K}]'$ and $\vec{M}'$ are strongly computable by Lemma 2. Hence $M[\vec{K}]'\vec{M}'$ is strongly computable and therefore strongly normalizable. In case it is an $\eta_\times$–conversion $M[\vec{K}]'$ and $N[\vec{K}]'$ are of the form $\pi_0(P)$ and $\pi_1(P)$, and the head conversion yields

$$\langle \pi_0(P), \pi_1(P)\rangle\vec{L}' \to P\vec{L}'.$$

By IH and Lemma 2 $\pi_0(P)$ and $\pi_1(P)$ are strongly computable, hence by definition also P. Since again by Lemma 2 $\vec{L}'$ is strongly computable we obtain that $P\vec{L}'$ is strongly computable and hence strongly normalizable. □

From Lemma 3 and Lemma 1 we immediately obtain

Theorem. *$\to$ is terminating, i.e. any term M is strongly normalizable.* □

Clearly the proof just given remains correct if we leave out the η–conversion rules (η) and $(\eta_\times)$. The corresponding normal form is called β–normal form.

This proof can be extended to terms involving primitive recursion operators (see e.g. [30, 31, 21]), the general recursion or fixed point operator (see [18]) or the bounded fixed point operator (see [26]).

4.2 Subformula property and Herbrand's Theorem

We now want to draw some conclusions from the fact that any derivation can be brought into β–normal form. For this purpose we have to analyse more closely the form of a normal derivation. First we need some terminological preparations.

With $p, q \in \{0,1\}^*$ we denote *positions* in a term M. $\mathrm{Pos}(M)$ is the set of all positions in M, and $[]$ denotes the empty list, i.e. the root position. $p0$ and $p1$ are the extensions of the position p by 0 or 1. The subterm of M at position p is denoted by M/p, and $M[N]_p$ results from M by replacing the subterm at position p by N. We write $p \leq q$ if p is a prefix of q. p and q are called *independent* (written $p \parallel q$) if $p \not\leq q$ and $q \not\leq p$. Firthermore let

$$\begin{aligned}
\mathrm{Leafpos}(M) &:= \{p \in \mathrm{Pos}(M) : M/p \text{ variable or constant}\},\\
\mathrm{Elimpos}(M) &:= \{p \in \mathrm{Pos}(M) : M/p \text{ formed by an elimination rule}\},\\
\mathrm{Intropos}(M) &:= \{p \in \mathrm{Pos}(M) : M/p \text{ formed by an introduction rule}\}.
\end{aligned}$$

For any $p \in \mathrm{Elimpos}(M) \cup \mathrm{Leafpos}(M)$ we define its associated minimal position by

$$\begin{aligned}
\mathrm{minpos}_M([]) &:= [].\\
\mathrm{minpos}_M(pi) &:= \begin{cases} \mathrm{minpos}_M(p), & \text{if } i = 0 \text{ and } p \in \mathrm{Elimpos}(M);\\ pi, & \text{if } i = 1 \text{ and } p \in \mathrm{Elimpos}(M), \text{ or}\\ & p \in \mathrm{Intropos}(M). \end{cases}
\end{aligned}$$

Clearly $\mathrm{minpos}_M(p) \in \mathrm{Elimpos}(M) \cup \mathrm{Leafpos}(M)$. Let

$$\mathrm{Minpos}(M) := \{p \in \mathrm{Elimpos}(M) \cup \mathrm{Leafpos}(M) : \mathrm{minpos}_M(p) = p\}.$$

For any $p \in \mathrm{Pos}(M)$ we define its associated end position by

$$\begin{aligned}
\mathrm{endpos}_M([]) &:= [].\\
\mathrm{endpos}_M(pi) &:= \begin{cases} \mathrm{endpos}_M(p), & \text{if } p \in \mathrm{Intropos}(M), \text{ or}\\ & i = 0 \text{ and } p \in \mathrm{Elimpos}(M);\\ pi, & \text{if } i = 1 \text{ and } p \in \mathrm{Elimpos}(M). \end{cases}
\end{aligned}$$

Let

$$\mathrm{Endpos}(M) := \{p \in \mathrm{Pos}(M) : \mathrm{endpos}_M(p) = p\}.$$

For β–normal terms M the leaf positions and the minimal positions are in a bijective correspondence. Note however that two minimal positions can yield the same end position, as e.g. in

$$\frac{P \quad Q}{P \wedge Q}$$

Let M be a term and $p \in \mathrm{Leafpos}(M)$. Then

$$\{q \in \mathrm{Pos}(M) : p \geq q \geq \mathrm{endpos}_M(p)\}$$

is called the *branch* in M determined by the leaf position p. In particular, $\mathrm{minpos}_M(p)$ is an element of the branch determined by p. In case $\mathrm{endpos}_M(p) = []$ the branch is called a *main branch*. In a β–normal term any branch $p_1 \geq \cdots \geq p_n$ has a particularly perspicious form: all elimination rules must come before all introduction rules.

In the case of a normal derivation term we want to draw from this some conclusions on the formulas attached to the positions. To do this we need the following notions.

The relation "A is a *subformula* of B" is the reflexive transitive closure of the relation "*immediate subformula*", defined as follows.

(i) A and B are immediate subformulas of $A \to B$,

(ii) A and B are immediate subformulas of $A \wedge B$,

(iii) $A[t/x]$ is an immediate subformula of $\forall x\, A$.

Furthermore we need the notion "A is a *strictly positive subformula* of B". This is to be the reflexive transitive closure of the relation "*immediate strictly positive subformula*", defined as follows.

(i) B is an immediate strictly positive subformula of $A \to B$,

(ii) A and B are immediate strictly positive subformulas of $A \wedge B$,

(iii) $A[t/x]$ is an immediate strictly positive subformula of $\forall x\, A$.

Now let d be a β–normal derivation term, $p_1 \geq \cdots \geq p_n$ a branch in d, $p_m = \mathrm{minpos}_d(p_1)$ and A_i the type of d/p_i. Then A_m must be a strictly positive subformula of all A_i with $i \neq m$. This A_m is called the *minimal formula* of the branch. Furthermore any A_i with $i \leq m$ is a strictly positive subformula of A_1, and any A_i with $i \geq m$ is a strictly positive subformula of A_n.

Theorem (subformula property). *If $d^B[u_1^{A_1}, \ldots, u_n^{A_n}]$ is a β–normal derivation and e^C is a subderivation of d^B, then C is a subformula of B or of some A_i.*

Proof by induction on the length of the end position of a branch in d. Here we need the properties of branches just mentioned. □

Theorem (Herbrand). *If $\forall \vec{x}_1 A_1, \ldots, \forall \vec{x}_m A_m \vdash B$ with quantifier free $A_1, \ldots, A_m, B$, then one can find terms $\vec{t}_{11}, \ldots, \vec{t}_{1n_1}, \ldots, \vec{t}_{m1}, \ldots, \vec{t}_{mn_m}$ such that*

$$A_1[\vec{t}_{11}], \ldots, A_1[\vec{t}_{1n_1}], \ldots, A_m[\vec{t}_{m1}], \ldots, A_m[\vec{t}_{mn_m}] \vdash B.$$

Proof. To simplify the notation we only treat the case $\forall x\, A \vdash B$ with quantifier free A, B. From the given derivation we can construct a normal derivation $d^B[u^{\forall x\, A}]$. By induction on the length of the end position of a branch it can be shown easily that any branch ends with a derivation of a quantifier free formula and begins with the rule $\forall^-$, i.e. with $u^{\forall x\, A} t_i$, or with a bound assumption variable v^C with quantifier free C. Now replace in the first case the subderivation $u^{\forall x\, A} t_i$ by $v_i^{A[t_i]}$ with a new assumption variable v_i. $\square$

4.3 Eta–expansion

We now define an η–expansion relation $\to_{\eta\text{–exp}}$ between terms. $M \to_{\eta\text{–exp}} M'$ is to mean that M' can be obtained from M by η–expansion of a subterm N, i.e. by replacing N by $\lambda x.Nx$ or by $\langle \pi_0(N), \pi_1(N)\rangle$ respectively, but only if no new β–convertible subterm is created. This can be ensured if η–expansions are allowed at minimal positions only. For instance a variable F of type $(\rho \to \rho) \to \rho$ can be η–expanded as follows:

$$F^{(\rho\to\rho)\to\rho} \to_{\eta\text{–exp}} \lambda f^{\rho\to\rho}.Ff \to_{\eta\text{–exp}} \lambda f.F(\lambda x.fx).$$

Definition. *$M \to_{\eta\text{–exp}} M'$ iff there is a $p \in \mathrm{Minpos}(M)$ such that the type of $N := M/p$ is composed and in case $\mathrm{typ}(N) = \rho \to \sigma$ we have*

$$M' = M[\lambda x^\rho.Nx]_p$$

with $x \notin \mathrm{FV}(N)$, and in case $\mathrm{typ}(N) = \rho \times \sigma$ we have

$$M' = M[\langle \pi_0(N), \pi_1(N)\rangle]_p.$$

Note that in case $M \to_{\eta\text{–exp}} M'$ the term M' again is β–normal if M is.

Definition. *A term M is in long normal form if it has no β–convertible or η–expandable subterm.*

Write $M \to_{\beta\eta\text{–exp}} M'$ if M' is obtained from M by β–conversion or (context dependent) η–expansion of exactly one subterm, i.e. if $M \to_\beta M'$ or $M \to_{\eta\text{–exp}} M'$, where $M \to_\beta M'$ denotes the one step relation defined using β–conversion rules only. Following Di Cosmo and Kesner [6],[7] we now prove that also $\to_{\beta\eta\text{–exp}}$ is terminating. A different proof based on an extension of Tait's method of computability predicates (and using ideas from Prawitz [19]) has been given by Dougherty [8], pages 142–146.

Note that the proof above cannot be extended to this situation, for the following reason. In Lemma 2 we have shown that from $M \to^* M'$ and strong computability of M we can infer strong computability of M'. In the proof we have made use of the fact that any reduction sequence starting with $M'\vec{N}$ can be extended to a reduction sequence starting with $M\vec{N}$. But this does not hold any more, since η–expansion is context depending.

The key idea of Di Cosmo's and Kesner's proof is the observation that η–expansion can be simulated by β–conversion:

$$(\lambda y^{\rho\to\sigma}\lambda x^\rho.yx)M \to_\beta \lambda x.Mx \quad \text{if } x \notin \mathrm{FV}(M),$$
$$(\lambda x^{\rho\times\sigma}\langle\pi_0(x),\pi_1(x)\rangle)M \to_\beta \langle\pi_0(M),\pi_1(M)\rangle.$$

More precisely, for any type ρ we define an η–expansor Δ_ρ by

$$\Delta_\iota := \lambda x^\iota\, x^\iota,$$
$$\Delta_{\rho\to\sigma} := \lambda y^{\rho\to\sigma}\lambda x^\rho.\Delta_\sigma(y\Delta_\rho x),$$
$$\Delta_{\rho\times\sigma} := \lambda x^{\rho\times\sigma}\langle\Delta_\rho\pi_0(x),\Delta_\sigma\pi_1(x)\rangle.$$

Note that Δ_ρ is a closed term of type $\rho\to\rho$. We write $\Delta^k_\rho M$ for $\Delta_\rho(\Delta_\rho(\dots(\Delta_\rho M)\dots))$ with k occurrences of Δ_ρ. The operator Δ^k is to have a higher precedence than application, so $\Delta^k MN$ means $(\Delta^k M)N$ and $\Delta^k M0$ means $(\Delta^k M)0$, i.e. $\pi_0(\Delta^k M)$. We write $\Delta^k\vec{M}$ for $\Delta^k M_1\dots\Delta^k M_n$.

Lemma.

$$\Delta^k M\vec{N} \to^*_\beta \Delta^k(M\Delta^k\vec{N}), \tag{1}$$
$$\Delta^k_{\rho\to\sigma}(\lambda x\, M) \to^*_\beta \lambda x.\Delta^k_\sigma(M[\Delta^k_\rho x/x]), \tag{2}$$
$$\Delta^k_{\rho\times\sigma}\langle M,N\rangle \to^*_\beta \langle\Delta^k_\rho M,\Delta^k_\sigma N\rangle. \tag{3}$$

Proof. Note that in (1) $\vec{N}$ is meant to be a list of terms or 0 or 1.

(1). We first assume that $\vec{N}$ has length 1. So we have to show

$$\Delta^k_{\rho\to\sigma}MN \to^*_\beta \Delta^k_\sigma(M\Delta^k_\rho N), \tag{1.1}$$
$$\pi_0(\Delta_{\rho\times\sigma}M) \to^*_\beta \Delta_\rho(\pi_0(M)), \tag{1.2}$$
$$\pi_1(\Delta_{\rho\times\sigma}M) \to^*_\beta \Delta_\sigma(\pi_1(M)). \tag{1.3}$$

We use induction on k. The case $k=0$ is trivial, and in the induction step we obtain for (1.1)

$$\begin{aligned}\Delta^{k+1}_{\rho\to\sigma}MN &= \Delta_{\rho\to\sigma}(\Delta^k_{\rho\to\sigma}M)N\\ &\to^*_\beta \Delta_\sigma(\Delta^k_{\rho\to\sigma}M\Delta_\rho N) \quad \text{by definition of } \Delta_{\rho\to\sigma}\\ &\to^*_\beta \Delta_\sigma(\Delta^k_\sigma(M\Delta^k_\rho(\Delta_\rho N))) \quad \text{by IH}\\ &= \Delta^{k+1}_\sigma(M\Delta^{k+1}_\rho N)\end{aligned}$$

and e.g. for (1.3)

$$\begin{aligned}\pi_1\left(\Delta^{k+1}_{\rho\times\sigma}M\right) &= \pi_1\left(\Delta_{\rho\times\sigma}(\Delta^k_{\rho\times\sigma}M)\right)\\ &\to^*_\beta \Delta_\sigma\pi_1\left(\Delta^k_{\rho\times\sigma}M\right) \quad \text{by definition of } \Delta_{\rho\times\sigma}\\ &\to^*_\beta \Delta_\sigma\left(\Delta^k_\sigma\pi_1(M)\right) \quad \text{by IH}\\ &= \Delta^{k+1}_\sigma\pi_1(M).\end{aligned}$$

We now prove (1) for an arbitrary $\vec{N}$, by induction on the length of $\vec{N}$. For empty $\vec{N}$ this is trivial, and for the induction step we obtain

$$\Delta^k_{\rho\to\sigma} M N \vec{N} \to^*_\beta \Delta^k_\sigma(M \Delta^k_\rho N)\vec{N} \to^*_\beta \Delta^k(M\Delta^k_\rho N \Delta^k \vec{N}) \quad \text{by IH.}$$

(2). We again use induction on k. The case $k = 0$ is trivial, and for the induction step we obtain

$$\begin{aligned}
\Delta^{k+1}_{\rho\to\sigma}(\lambda x\, M) &= \Delta_{\rho\to\sigma}(\Delta^k_{\rho\to\sigma}(\lambda x\, M)) \\
&\to^*_\beta \lambda x\, \Delta_\sigma(\Delta^k_{\rho\to\sigma}(\lambda x\, M)\Delta_\rho x) \quad \text{by definition of } \Delta_{\rho\to\sigma} \\
&\to^*_\beta \lambda x\, \Delta_\sigma(\Delta^k_\sigma(M[\Delta^k_\rho x/x])[\Delta_\rho x/x]) \quad \text{by IH} \\
&= \lambda x\, \Delta^{k+1}_\sigma M[\Delta^{k+1}_\rho x/x].
\end{aligned}$$

(3). Similar. □

By a Δ–*expansion* of M we mean a term M^Δ obtained from M by replacing simultaneously every η–expandable subterm N of type ρ in M by $\Delta^k_\rho N$ for some $k > 0$.

By an *inner* Δ–*expansion* of M we mean a term $M^{i\Delta}$ obtained from M by replacing simultaneously every η–expandable subterm N of type ρ in M exept the term M itself by $\Delta^k_\rho N$ for some $k > 0$.

By a *strong* Δ–*expansion* of M we mean a term $M^{s\Delta}$ obtained from M by replacing simultaneously every η–expandable subterm N of type ρ in M by $\Delta^k_\rho N$ for some $k > 0$ and every other subterm N of type ρ in M by $\Delta^k_\rho N$ for some $k \geq 0$.

Lemma. *Let $M^{s\Delta}$ be a strong Δ–expansion of M. Then we can find a Δ–expansion M^Δ of M such that $M^{s\Delta}$ β–reduces to M^Δ.*

Proof by induction on M, using the previous Lemma.

Case $\lambda x\, M$. Any $(\lambda x\, M)^{s\Delta}$ has the form $\Delta^k(\lambda x\, M^{s\Delta})$ with $k \geq 0$. By the previous Lemma $\Delta^k(\lambda x\, M^{s\Delta})$ β–reduces to $\lambda x.\Delta^k(M^{s\Delta}[\Delta^k x/x])$. By IH($M$) we can find a Δ–expansion M^Δ of M such that $\Delta^k(M^{s\Delta}[\Delta^k x/x])$ β–reduces to M^Δ. Let $(\lambda x\, M)^\Delta := \lambda x\, M^\Delta$.

Case $\langle M, N\rangle$. Similar.

Case $M\vec{N}$, M an introduction and $\vec{N}$ not empty. Any $(M\vec{N})^{s\Delta}$ must have the form $\Delta^k(\Delta^m M^{s\Delta}\vec{N}^{s\Delta})$ with $k > 0$, $m \geq 0$ and $M^{s\Delta}$ an introduction. By the previous Lemma $\Delta^m M^{s\Delta}\vec{N}^{s\Delta}$ β–reduces to $\Delta^m(M^{s\Delta}\Delta^m\vec{N}^{s\Delta})$. By IH($M$) we can find a Δ–expansion M^Δ of M such that $M^{s\Delta}$ β–reduces to M^Δ. By IH(N_i) we can find a Δ–expansion N_i^Δ of N_i such that $\Delta^m N_i^{s\Delta}$ β–reduces to N_i^Δ. Let $(M\vec{N})^\Delta := \Delta^{k+m}(M^\Delta\vec{N}^\Delta)$.

Case $x\vec{N}$. Any $(x\vec{N})^{s\Delta}$ has the form $\Delta^k(\Delta^m x\vec{N}^{s\Delta})$ with $k > 0$ and $m \geq 0$. By the previous Lemma $\Delta^m x\vec{N}^{s\Delta}$ β–reduces to $\Delta^m(x(\Delta^m\vec{N}^{s\Delta}))$. By IH($N_i$) we can find an Δ–expansion N_i^Δ of N_i such that $\Delta^m N_i^{s\Delta}$ β–reduces to N_i^Δ. Let $(x\vec{N})^\Delta := \Delta^{k+m}(x\vec{N}^\Delta)$. □

For the proof of the following theorem we need an inductive definition of η–expansion, which can be given as follows.

Definition. *$M \to_{0\eta-\mathrm{exp}} M'$ is defined by*

$$
\begin{aligned}
&M \to_{0\eta-\mathrm{exp}} \lambda x.Mx \quad \textit{if } M \textit{ is not of the form } \lambda y\, N,\\
&M \to_{0\eta-\mathrm{exp}} \langle \pi_0(M), \pi_1(M)\rangle \quad \textit{if } M \textit{ is not a pair.}
\end{aligned}
$$

From $\to_{0\eta-\mathrm{exp}}$ one derives a one–step η–expansion relation $\to_{\eta-\mathrm{exp}}$ as follows.

Definition. *$M \to_{\eta-\mathrm{exp}} M'$ is defined inductively by*

$$
\begin{aligned}
M \to_{0\eta-\mathrm{exp}} M' &\implies M \to_{\eta-\mathrm{exp}} M'.\\
M \to_{\eta-\mathrm{exp}} M' &\implies (\lambda x\, M)\vec{K} \to_{\eta-\mathrm{exp}} (\lambda x\, M')\vec{K}.\\
M \to_{\eta-\mathrm{exp}} M' &\implies \langle M, N\rangle\vec{K} \to_{\eta-\mathrm{exp}} \langle M', N\rangle\vec{K}.\\
N \to_{\eta-\mathrm{exp}} N' &\implies \langle M, N\rangle\vec{K} \to_{\eta-\mathrm{exp}} \langle M, N'\rangle\vec{K}.\\
N \to_{\eta-\mathrm{exp}} N' &\implies MN\vec{K} \to_{\eta-\mathrm{exp}} MN'\vec{K}.
\end{aligned}
$$

Note that a similar definition of $\to_\beta$ can be given, as follows:

Definition. *$M \to_\beta M'$ is defined inductively by*

$$
\begin{aligned}
M \to_{0\beta} M' &\implies M\vec{N} \to_\beta M'\vec{N}.\\
M \to_\beta M' &\implies (\lambda x\, M)\vec{K} \to_\beta (\lambda x\, M')\vec{K}.\\
M \to_\beta M' &\implies \langle M, N\rangle\vec{K} \to_\beta \langle M', N\rangle\vec{K}.\\
N \to_\beta N' &\implies \langle M, N\rangle\vec{K} \to_\beta \langle M, N'\rangle\vec{K}.\\
N \to_\beta N' &\implies MN\vec{K} \to_\beta MN'\vec{K}.
\end{aligned}
$$

This remark will be used in the proof of the theorem below.

Theorem. *If $M \to_{\beta\eta-\mathrm{exp}} N$, then for any Δ–expansion M^Δ of M we can find a Δ–expansion N^Δ of N such that N^Δ can be obtained from M^Δ by a non–empty sequence of β–reductions.*

Proof by induction on the definition of $M \to_{\beta\eta-\mathrm{exp}} N$. Recall $\to_{\beta\eta-\mathrm{exp}} = \to_\beta \cup \to_{\eta-\mathrm{exp}}$. We only treat the cases concerning types $\rho \to \sigma$; the cases concerning product types $\rho \times \sigma$ can be dealt with in a similar fashion.

Case $(\lambda x\, M)N\vec{N} \to_\beta M[N/x]\vec{N}$. Any Δ–expansion of $(\lambda x\, M)N\vec{N}$ must be of the form $\Delta^k((\lambda x\, M^\Delta)N^\Delta\vec{N}^\Delta)$ for some $k > 0$ with Δ–expansions M^Δ of M, N^Δ of N and

$\vec{N}^\Delta$ of $\vec{N}$. One β–reduction yields $\Delta^k(M^\Delta[N^\Delta/x]\vec{N}^\Delta)$. Since this term is a strong Δ–expansion of $M[N/x]\vec{N}$ we can β–reduce it by the previous Lemma to a Δ–expansion of $M[N/x]\vec{N}$.

Case $M \to_{0\eta-\mathrm{exp}} \lambda x.Mx$. Then M is not a λ–expression. Hence any Δ–expansion of M must be of the form $\Delta^{k+1}M^{i\Delta}$ with an inner Δ–expansion $M^{i\Delta}$ of M. From $\Delta = \lambda y \lambda x.\Delta(y\Delta x)$ we obtain immediately

$$\Delta^{k+1}M^{i\Delta} = \Delta^k(\Delta M^{i\Delta}) \to_\beta \Delta^k\lambda x.\Delta(M^{i\Delta}\Delta x).$$

Since this term is a strong Δ–expansion of $\lambda x.Mx$ we can β–reduce it by the previous Lemma to a Δ–expansion of $\lambda x.Mx$.

In the following cases let $\to$ denote either $\to_{\eta-\mathrm{exp}}$ or else $\to_\beta$.

Case $(\lambda x\, M)\vec{K} \to (\lambda x\, M')\vec{K}$. Any Δ–expansion of $(\lambda x\, M)\vec{K}$ must be of the form $\Delta^k((\lambda x\, M^\Delta)\vec{K}^\Delta)$ for some $k > 0$ with Δ–expansions M^Δ of M and $\vec{K}^\Delta$ of $\vec{K}$. By IH we can find a Δ–expansion M'^Δ of M' such that M'^Δ can be obtained from M^Δ by a non–empty sequence of β–reductions. Let $((\lambda x\, M')\vec{K})^\Delta := \Delta^k((\lambda x\, M'^\Delta)\vec{K}^\Delta)$.

Case $x\vec{M}N\vec{K} \to x\vec{M}N'\vec{K}$. Then any Δ–expansion of $x\vec{M}N\vec{K}$ must be of the form $\Delta^k(x\vec{M}^\Delta N^\Delta \vec{K}^\Delta)$ for some $k > 0$ with Δ–expansions $\vec{M}^\Delta$ of $\vec{M}$, N^Δ of N and $\vec{K}^\Delta$ of $\vec{K}$. By IH we can find a Δ–expansion N'^Δ of N' such that N'^Δ can be obtained from N^Δ by a non–empty sequence of β–reductions. Let $(x\vec{M}N'\vec{K})^\Delta := \Delta^k(x\vec{M}^\Delta N'^\Delta \vec{K}^\Delta)$.

Case $(\lambda x\, M)\vec{M}N\vec{K} \to (\lambda x\, M)\vec{M}N'\vec{K}$. Any Δ–expansion of $(\lambda x\, M)\vec{M}N\vec{K}$ must be of the form $\Delta^k((\lambda x\, M^\Delta)\vec{M}^\Delta N^\Delta \vec{K}^\Delta)$ for some $k > 0$ with Δ–expansions M^Δ of M, $\vec{M}^\Delta$ of $\vec{M}$, N^Δ of N and $\vec{K}^\Delta$ of $\vec{K}$. By IH we can find a Δ–expansion N'^Δ of N' such that N'^Δ can be obtained from N^Δ by a non–empty sequence of β–reductions. Let $((\lambda x\, M)\vec{M}N'\vec{K})^\Delta := \Delta^k((\lambda x\, M^\Delta)\vec{M}^\Delta N'^\Delta \vec{K}^\Delta)$. □

Corollary. $\to_{\beta\eta-\mathrm{exp}}$ *is terminating.* □

4.4 Minimal from classical proofs

We now use the long normal form of derivation terms to show that any classical proof of $\perp$ from so–called generalized (definite) Horn formulas can be converted into a proof in minimal logic, Moreover we describe a reasonable algorithm to do this conversion; here we follow [22]. This result can be used to prove completeness of SLD–Resolution.

A formula is called *Horn formula* if it has the form $\forall x_1, \dots, x_n.A_1 \to \dots \to A_m \to B$ with A_i and B atomic. It is called *definite Horn formula* if in addition we have $B \neq \perp$. If instead of atomic A_i we allow universally quantified atomic formulas, the result is called a *generalized (definite) Horn formula*.

Theorem 1. *Let $A_1, \dots, A_n$ be generalized Horn formulas. We have a quadratic algorithm transforming a classical proof in long normal form of $\bot$ from $A_1, \dots, A_n$ into a proof in minimal logic of $\bot$ from the same assumptions.*

Proof by induction on the total number of stability axioms used. Note first that bound assumption variables u in the given normal proof can only occur in the context

$$\mathrm{Stab}_R^{\forall \vec{x}. \neg\neg R(\vec{x}) \to R(\vec{x})} \vec{t}(\lambda u\, d)$$

with u of type $\neg R(\vec{t})$ and d of type $\bot$. The reason for this is that all top formulas different from stability axioms are generalized Horn formulas which never have an implication in the premise of another implication.

Case 1. There is at least one occurrence of a bound assumption variable in the proof. Since we assume our proof to be in long normal form, any of the occurrences of an assumption variable u of type $\neg R(\vec{t})$ must be the main premise of an $\to$-elimination, i.e. must be in a context ud_1 where d_1 derives $R(\vec{t})$. Now choose an uppermost occurrence of a bound assumption variable, i.e. a subderivation ud_1 where d_1 does not contain an occurrence of any bound assumption variable. Since d_1 derives $R(\vec{t})$, we can replace the whole subderivation $\mathrm{Stab}_R(\vec{t})(\lambda u\, d)$ of $R(\vec{t})$ (the one where u is bound) by d_1. Hence we have removed one occurrence of a stability axiom.

Case 2. Otherwise. If there are no more stability axioms in the proof, we are done. If not, choose an uppermost occurrence of a stability axiom, i.e. a subderivation $\mathrm{Stab}_R(\vec{t})(\lambda u\, d)$ where d does not contain stability axioms. Since we are in case 2 here d also cannot contain free assumption variables which are bound elsewhere in the proof. But since d derives $\bot$, we can replace the whole proof (which also has $\bot$ as its end formula) by d and hence we are done again. □

Note that Theorem 1 is best possible in the sense that it becomes false if we allow an implication in the body of one of the Horn formulas. A counterexample (due to U. Berger) is

$$((P \to Q) \to \bot) \to (P \to \bot) \to \bot,$$

which is provable in classical but not in minimal logic. For if it were, we could replace $\bot$ in this proof (which in minimal logic is just another propositional variable) by P, and hence we would obtain a proof in minimal logic of the Peirce formula

$$((P \to Q) \to P) \to P,$$

which is known to be underivable.

By essentially the same argument we obtain the following variant of Theorem 1 for generalized *definite* Horn formulas:

Theorem 2. *Let $A_1, \dots, A_n$ be generalized definite Horn formulas. We have a quadratic algorithm transforming a classical proof in long normal form of an atomic formula B from $A_1, \dots, A_n$ into a proof in minimal logic of B from the same assumptions.*

Proof by a simple modification of the argument for Theorem 1. Note that in case 2 it cannot happen that stability axioms occur in the proof since then we would have a derivation d of $\perp$ from definite Horn formulas, which is clearly impossible. $\square$

4.5 Uniqueness of normal forms

We now show the uniqueness of normal forms. More precisely, we show that the one–step relations $\rightarrow_{\beta\eta}$ and $\rightarrow_{\beta\eta-\mathrm{exp}}$ are confluent and hence that the $\beta\eta$ normal form as well as the long normal form are uniquely determined. This follows from the local confluence and termination of these relations by the Lemma of Newman [15].

Remark: If we leave out the conversion rule

$$\langle \pi_0(M), \pi_1(M) \rangle \rightarrow_0 M \tag{$\eta_\times$}$$

(expressing surjectivity of pairing), then there is an elegant alternative method to prove confluence of $\rightarrow_{\beta\eta}$, which does not use termination. The idea of this proof goes back to J.B. Rosser and W.W. Tait and uses a parallel reduction relation $\rightarrow_{\mathrm{par}}$ satisfying $\rightarrow_{\beta\eta} \subseteq \rightarrow_{\mathrm{par}} \subseteq \rightarrow^*_{\beta\eta}$. For this relation the diamond property

$$M \rightarrow_{\mathrm{par}} M', M \rightarrow_{\mathrm{par}} M'' \implies \exists M'''.M' \rightarrow_{\mathrm{par}} M''', M'' \rightarrow_{\mathrm{par}} M'''$$

can be proved. For $\rightarrow_{\beta\eta}$ instead of $\rightarrow_{\mathrm{par}}$ this property does not hold. The reason that it holds for $\rightarrow_{\mathrm{par}}$ is that $\rightarrow_{\mathrm{par}}$ is compatible with substitution in the following sense: we have

$$M \rightarrow_{\mathrm{par}} M', N \rightarrow_{\mathrm{par}} N' \implies M[N] \rightarrow_{\mathrm{par}} M'[N'].$$

An exposition can be found in [25].

Let $\rightarrow$ be the one–step relation $\rightarrow_{\beta\eta}$. For the proof of local confluence of $\rightarrow$ we need the following fact concerning compatibility of $\rightarrow$ with substitution.

Lemma. *(i)* $M \rightarrow M' \implies M[N] \rightarrow M'[N]$.
(ii) $N \rightarrow N' \implies M[N] \rightarrow^* M[N']$. $\square$

Note that (ii) with $\rightarrow$ instead of $\rightarrow^*$ does not hold. The reason is that the variable x to be substituted can have multiple occurrences in M, while the relation $M \rightarrow M'$ means that M' is obtained from M by conversion at *exactly one* position.

Lemma. $\rightarrow_{\beta\eta}$ *is locally confluent, i.e.*

$$M \rightarrow M', M \rightarrow M'' \implies \exists M'''.M' \rightarrow^* M''', M'' \rightarrow^* M'''.$$

Proof by induction on M. If $M \rightarrow M'$ and $M \rightarrow M''$ are obtained by the same rule the claim follows immediately from the IH. The remaining cases split into two groups, depending on whether M is an abstraction or an application, or a pair or a component, respectively. We start with the first group of cases.

Case $-_0, -_1$.

$$\begin{array}{ccc} MN & \to & M'N \\ \downarrow & & \downarrow \\ MN' & \to & M'N' \end{array}$$

Case $-_0+, \beta$.

$$\begin{array}{ccc} (\lambda x\, M)N & \to & (\lambda x\, M')N \\ \downarrow & & \downarrow \\ M[N] & \to & M'[N] \end{array}$$

Here we have used $M \to M' \implies M[N] \to M'[N]$.

Case $-_0\eta, \beta$.

$$\begin{array}{ccc} (\lambda x.Mx)N & \to & MN \\ \downarrow & & = \\ MN & = & MN \end{array}$$

Case $-_1, \beta$.

$$\begin{array}{ccc} (\lambda x\, M)N & \to & (\lambda x\, M)N' \\ \downarrow & & \downarrow \\ M[N] & \to^* & M[N'] \end{array}$$

Here we have used $N \to N' \implies M[N] \to^* M[N']$.

Case $+-_0, \eta$.

$$\begin{array}{ccc} \lambda x.Mx & \to & \lambda x.M'x \\ \downarrow & & \downarrow \\ M & \to & M' \end{array}$$

Case $+-_1, \eta$. This case cannot occur, since there is no N such that $x \to N$.

Case $+\beta, \eta$.

$$\begin{array}{ccc} \lambda x.(\lambda x\, M)x & \to & \lambda x\, M \\ \downarrow & & = \\ \lambda x\, M & = & \lambda x\, M \end{array}$$

We now treat the group of cases concerning pairs and components. We leave out the index $\times$.

Case $-_0, -_1$.

$$\begin{array}{ccc} \langle M, N\rangle & \to & \langle M', N\rangle \\ \downarrow & & \downarrow \\ \langle M, N'\rangle & \to & \langle M', N'\rangle \end{array}$$

Case $-_0+_0, \beta_0$ (and similarly $-_1+_1, \beta_1$).

$$\begin{array}{ccc} \pi_0\langle M_0, M_1\rangle & \to & \pi_0\langle M_0', M_1\rangle \\ \downarrow & & \downarrow \\ M_0 & \to & M_0' \end{array}$$

Case $-_0+_1, \beta_0$ (and similarly $-_1+_0, \beta_1$).

$$\begin{array}{ccc} \pi_0\langle M_0, M_1\rangle & \to & \pi_0\langle M_0, M_1'\rangle \\ \downarrow & & \downarrow \\ M_0 & = & M_0 \end{array}$$

Case $-_i\eta, \beta_i$.

$$\begin{array}{ccc} \pi_i\langle \pi_0(M), \pi_1(M)\rangle & \to & \pi_i(M) \\ \downarrow & & = \\ \pi_i(M) & = & \pi_i(M) \end{array}$$

Case $+_0-_0, \eta$ (and similarly $+_1-_1, \eta$).

$$\begin{array}{ccc} \langle \pi_0(M), \pi_1(M)\rangle & \to & \langle \pi_0(M'), \pi_1(M)\rangle \\ \downarrow & & \downarrow_* \\ M & \to & M' \end{array}$$

Case $+_0\beta_0, \eta$ (and similarly $+_1\beta_1, \eta$).

$$\begin{array}{ccc} \langle \pi_0\langle M_0, M_1\rangle, \pi_1\langle M_0, M_1\rangle\rangle & \to & \langle M_0, \pi_1\langle M_0, M_1\rangle\rangle \\ \downarrow & & \downarrow \\ \langle M_0, M_1\rangle & = & \langle M_0, M_1\rangle \end{array}$$

$\square$

Lemma of Newman. *If $\to$ is locally confluent and terminating, then $\to$ is confluent, i.e.*

$$M \to^* M', M \to^* M'' \implies \exists M'''.M' \to^* M''', M'' \to^* M'''.$$

Proof. A term M is called *good* if it satisfies the property of confluence, i.e. if for all M', M'' we have

$$M \to^* M', M \to^* M'' \implies \exists M'''.M' \to^* M''', M'' \to^* M''';$$

otherwise M is called *ambiguous*. Now assume that an ambiguous term M exists. We show that then there is an ambiguous term N such that $M \to^+ N$. This contradicts our assumption that $\to$ is terminating.

So let M be ambiguous, and M', M'' be terms satisfying $M \to^* M'$ and $M \to^* M''$ and such that there is no term M''' satisfying $M' \to^* M'''$ and $M'' \to^* M'''$. We then have M_1, M_2 such that $M \to M_1 \to^* M'$ and $M \to M_2 \to^* M''$. From the local confluence of $\to$ we obtain an M_3 such that

$$\begin{tikzcd} M \arrow[r] \arrow[d] & M_1 \arrow[r, "*"] \arrow[d, "*"] & M' \\ M_2 \arrow[r, "*"] \arrow[d, "*"] & M_3 & \\ M'' & & \end{tikzcd}$$

Now if M_1, M_2, M_3 were all good, we would have terms N, K, L such that

$$\begin{tikzcd} M \arrow[r] \arrow[d] & M_1 \arrow[r, "*"] \arrow[d, "*"] & M' \arrow[d, "*"] \\ M_2 \arrow[r, "*"] \arrow[d, "*"] & M_3 \arrow[r, "*"] \arrow[d, "*"] & N \arrow[d, "*"] \\ M'' \arrow[r, "*"] & K \arrow[r, "*"] & L \end{tikzcd}$$

But this contradicts our assumption that M is ambiguous. □

Corollary (Church–Rosser). $M =_{\beta\eta} N \iff \exists K.M \to^* K, N \to^* K.$

Proof: $\Longleftarrow$. Clear. $\Longrightarrow$. Induction on the definition of $M =_{\beta\eta} N$. □

Now let $\to$ be the one–step relation $\to_{\beta\eta^-\mathrm{exp}}$. For the proof of local confluence of $\to$ we need the following fact concerning compatibility of $\to$ with substitution.

Lemma. *(i)* $M \to M' \implies M[N] \to M'[N]$.

(ii) $N \to N' \implies \exists K. M[N], M[N'] \to^* K$.

Proof. (ii) If $N \to N'$ is not a head η–expansion we have $M[N] \to^* M[N']$ and are done. Otherwise, i.e. if $N' = \lambda z.Nz$ or $N' = \langle \pi_0(N), \pi_1(N) \rangle$, we generally have $M[N] \not\to^* M[N']$, since η–expansion is context dependent. But in this case we can find a K such that $M[N], M[N'] \to^* K$. To see this consider the occurrences of x in M. If x is in a context xL with L term or 0 or 1 replace xL by NL; otherwise replace x by $\lambda z.Nz$ or $\langle \pi_0(N), \pi_1(N) \rangle$, respectively. □

Lemma. $\to$ *is locally confluent, i.e.*

$$M \to M', M \to M'' \implies \exists M'''. M' \to^* M''', M'' \to^* M'''.$$

Proof. We argue as before in the case $\to_{\beta\eta}$, by induction on M. If $M \to M'$ and $M \to M''$ are obtained by the same rule or $M \to M'$ or $M \to M''$ are formed by head η–expansion, the claim follows immediately from the IH. The remaining cases split into two groups, depending on whether M is an abstraction or an application, or a pair or a component, respectively. We start with the first group of cases.

Case $-_0, -_1$.

$$\begin{array}{ccc} MN & \to & M'N \\ \downarrow & & \downarrow \\ MN' & \to & M'N' \end{array}$$

Case $-_0+, \beta$.

$$\begin{array}{ccc} (\lambda x\, M)N & \to & (\lambda x\, M')N \\ \downarrow & & \downarrow \\ M[N] & \to & M'[N] \end{array}$$

Here we have used $M \to M' \implies M[N] \to M'[N]$.

Case $-_1, \beta$.

$$\begin{array}{ccc} (\lambda x\, M)N & \to & (\lambda x\, M)N' \\ \downarrow & & \downarrow_* \\ M[N] & \to^* & K \end{array}$$

Here we have used $N \to N' \implies \exists K. M[N], M[N'] \to^* K$.

We now treat the group of cases concerning pairs and components. We leave out the index $\times$.

Case $-_0, -_1$.

$$\begin{array}{ccc}
\langle M, N\rangle & \to & \langle M', N\rangle \\
\downarrow & & \downarrow \\
\langle M, N'\rangle & \to & \langle M', N'\rangle
\end{array}$$

Case $-_0+_0, \beta_0$ (and similarly $-_1+_1, \beta_1$).

$$\begin{array}{ccc}
\pi_0\langle M_0, M_1\rangle & \to & \pi_0\langle M_0', M_1\rangle \\
\downarrow & & \downarrow \\
M_0 & \to & M_0'
\end{array}$$

Case $-_0+_1, \beta_0$ (and similarly $-_1+_0, \beta_1$).

$$\begin{array}{ccc}
\pi_0\langle M_0, M_1\rangle & \to & \pi_0\langle M_0, M_1'\rangle \\
\downarrow & & \downarrow \\
M_0 & = & M_0
\end{array}$$

$\square$

4.6 Normalization by evaluation

We now show that normalization can be achived by evaluation, following [2]. We make use of the fact proved above that any term M has a unique long normal form which we now denote by $\mathrm{lnf}(M)$. The following properties of the long normal form will be used.

$$M =_{\beta\eta} \mathrm{lnf}(M), \tag{1}$$
$$M =_{\beta\eta} M' \implies \mathrm{lnf}(M) = \mathrm{lnf}(M'), \tag{2}$$
$$\mathrm{lnf}(Mx) = N \implies \mathrm{lnf}(M) = \lambda x\, N \quad \text{if } x \notin \mathrm{FV}(M), \tag{3}$$
$$\mathrm{lnf}(M0) = N_0 \text{ and } \mathrm{lnf}(M1) = N_1 \implies \mathrm{lnf}(M) = \langle N_0, N_1\rangle, \tag{4}$$
$$\mathrm{lnf}(\vec{N}) = \vec{K} \text{ and } x\vec{N} \text{ of ground type } \implies \mathrm{lnf}(x\vec{N}) = x\vec{K}. \tag{5}$$

A term M is in long normal form if $\mathrm{lnf}(M) = M$. Let

$T_\rho =$ the set of all terms of type ρ,
$L_\rho =$ the set of all terms of type ρ in long normal form,
$A_\rho =$ the set of all terms of type ρ of the form $x\vec{N}$ with $\vec{N}$ in long normal form.

We define a model $(D_\rho)_\rho$ of our language as follows. For any ground type ι let D_ι be the set L_ι of all terms of type ι in long normal form. Furthermore let $D_{\rho\to\sigma}$ be the set of all functions $f\colon D_\rho \to D_\sigma$ and $D_{\rho\times\sigma}$ be the cartesian product $D_\rho \times D_\sigma$.

For any type ρ we define functions $\varphi_\rho \colon D_\rho \to L_\rho$ and $\psi_\rho \colon A_\rho \to D_\rho$ by simultaneous recursion on ρ. To see that φ_ρ and ψ_ρ are well–defined we need the properties (1)–(5) of the long normal form listed above.

$$\begin{aligned}
&\varphi_\iota(M) = M,\\
&\varphi_{\rho\to\sigma}(f) = \lambda x.\varphi_\sigma(f(\psi_\rho(x))), \quad x \text{ a new variable},\\
&\varphi_{\rho\times\sigma}([a,b]) = \langle \varphi_\rho(a), \varphi_\sigma(b)\rangle
\end{aligned}$$

and

$$\begin{aligned}
&\psi_\iota(x\vec{M}) = x\vec{M},\\
&\psi_{\rho\to\sigma}(x\vec{M})(a) = \psi_\sigma(x\vec{M}\varphi_\rho(a)),\\
&\psi_{\rho\times\sigma}(x\vec{M}) = [\psi_\rho(x\vec{M}0), \psi_\sigma(x\vec{M}1)].
\end{aligned}$$

This definition of ψ_ρ can be condensed into

$$\psi_\rho(x\vec{M})(\vec{a}) = x\vec{M}\varphi(\vec{a}).$$

Similar to Tait's notion of strong computability we define for any type ρ a relation R_ρ on $T_\rho \times D_\rho$ by

$$\begin{aligned}
&R_\iota(M,a) \iff \mathrm{lnf}(M) = a,\\
&R_{\rho\to\sigma}(M,f) \iff \forall N, a.R_\rho(N,a) \implies R_\sigma(MN, f(a)),\\
&R_{\rho\times\sigma}(M,[a,b]) \iff R_\rho(M0,a) \text{ and } R_\sigma(M1,b).
\end{aligned}$$

Note that $M =_{\beta\eta} M'$ implies $R(M,a) \iff R(M',a)$; this follows from (2) by induction on M.

Lemma 1. *(i)* $R_\rho(M,a) \implies \mathrm{lnf}(M) = \varphi_\rho(a)$.
(ii) $\vec{M}$ *in long normal form* $\implies R_\rho(x\vec{M}, \psi_\rho(x\vec{M}))$.

Proof. Induction on ρ. *Case* ι. Clear.

Case $\rho \to \sigma$, (i). Assume $R_{\rho\to\sigma}(M,f)$. We must show $\mathrm{lnf}(M) = \lambda x\, \varphi_\sigma(f(\psi_\rho(x)))$. By (3) it suffices to show that $\mathrm{lnf}(Mx) = \varphi_\sigma(f(\psi_\rho(x)))$. By IH_σ(i) it suffices to show $R_\sigma(Mx, f(\psi_\rho(x)))$. By definition of $R_{\rho\to\sigma}(M,f)$ it suffices to show $R_\rho(x, \psi_\rho(x))$. But this holds by IH_ρ(ii).

Case $\rho \times \sigma$, (i). Assume $R_{\rho\times\sigma}(M,[a,b])$, hence $R_\rho(M0,a)$ and $R_\sigma(M1,b)$. We must show $\mathrm{lnf}(M) = \langle\varphi_\rho(a), \varphi_\sigma(b)\rangle$. By (4) it suffices to show $\mathrm{lnf}(M0) = \varphi_\rho(a)$ and $\mathrm{lnf}(M1) = \varphi_\sigma(b)$. But this follows from IH_ρ(i) and IH_σ(i).

Case (ii). Let $\vec{M}$ be in long normal form. We must show $R_\rho(x\vec{M}, \psi_\rho(x\vec{M}))$. So let $\vec{N}$, $\vec{a}$ mit $R(\vec{N}, \vec{a})$ be given. We must show $\mathrm{lnf}(x\vec{M}\vec{N}) = x\vec{M}\varphi(\vec{a})$. By (5) it suffices to show $\mathrm{lnf}(\vec{N}) = \varphi(\vec{a})$. But this follows from IH(i). □

Since in our model $D_{\rho\to\sigma}$ is the set of all functions $f \colon D_\rho \to D_\sigma$ and $D_{\rho\times\sigma}$ is the cartesian product $D_\rho \times D_\sigma$ we can define the value $[\![M]\!]_U \in D_\rho$ for any term M of type ρ and any environment U in the standard way (cf. §1).

Lemma 2. *Assume $R(c, [\![c]\!])$ for any constant c. Then for any term $M[\vec{x}]$ of type ρ and any environment U*

$$R(\vec{K}, U(\vec{x})) \implies R(M[\vec{K}/\vec{x}], [\![M]\!]_U).$$

Proof by induction on M. *Case* x_i. Clear. *Case* c. By assumption.

Case $\lambda x\, M$. Let N, a be given such that $R_\rho(N, a)$. We must show

$$R_\sigma((\lambda x\, M[\vec{K}/\vec{x}])N, [\![\lambda x\, M]\!]_U(a)).$$

Since by the remark above R_σ is compatible with $=_{\beta\eta}$ it suffices to show

$$R_\sigma(M[N, \vec{K}/x, \vec{x}], [\![M]\!]_{U_x^a}).$$

But this holds by IH for M.

Case MN. We must show

$$R((MN)[\vec{K}/\vec{x}], [\![MN]\!]_U).$$

By definition of R it suffices to show $R(M[\vec{K}/\vec{x}], [\![M]\!]_U)$ and $R(N[\vec{K}/\vec{x}], [\![N]\!]_U)$. But this holds by IH for M and N.

Case $\langle M, N\rangle$. We must show $R_\rho(M[\vec{K}/\vec{x}], [\![M]\!]_U)$ and $R_\sigma(N[\vec{K}/\vec{x}], [\![N]\!]_U)$. But this holds by IH for M and N.

Case Mi. We must show $R((Mi)[\vec{K}/\vec{x}], [\![Mi]\!]_U)$. By definition of R it suffices to show $R(M[\vec{K}/\vec{x}], [\![M]\!]_U)$. But this holds by IH for M. $\square$

Now consider the special environment $U_\psi(x^\sigma) := \psi_\sigma(x)$. We write $[\![M]\!]_\psi$ for $[\![M]\!]_{U_\psi}$. Then we have $R(x, U_\psi(x))$ by Lemma 1(ii) and hence by Lemma 2 $R(M, [\![M]\!]_\psi)$ for any term M. Hence by Lemma 1(i) $\varphi_\rho([\![M]\!]_\psi)$ is the long normal form of M.

Remark. This observation leads to an efficient implementation of normalization for terms with free variables. The φ_ρ and ψ_ρ can be computed easily, and one can use the built–in evaluation mechanism of Scheme to provide the value $[\![M]\!]_\psi$ of a term M in the environment U_ψ.

4.7 Normal versus non–normal derivations

We now show that the requirement to give a normal derivation of a derivable formula can sometimes be unrealistic. Following Orevkov [16] we give examples of formulas C_k which are easily derivable with non–normal derivations (whose number of nodes is linear in k), but which require a non–elementary (in k) number of nodes in any normal derivation.

The example is related to Gentzens proof in [12] of transfinite induction up to ω_k in arithmetic; see e.g. [20] for an exposition. There the function $y \oplus \omega^x$ plays a crucial role, and also the assignment of a "lifting"–formula A^+ to any formula A, by

$$A^+ := \forall y.(\forall z \prec y)A[z/x] \to (\forall z \prec y \oplus \omega^x)A[z/x].$$

Here we consider the numerical function $y + 2^x$ instead, and axiomatize its graph by means of Horn clauses. The formula C_k expresses that from these axioms the existence of 2_k follows. A short, non–normal proof of this fact can then be given by a modification of Gentzen's idea, and it is easily seen that any normal proof of C_k must contain at least 2_k nodes.

The derivations to be given make heavy use of the existential quantifier $\exists$ defined by $\neg\forall\neg$. In particular we need:

Existence–Introduction–Lemma. $\vdash A \to \exists x\, A$.

Proof. A derivation term is

$$\lambda u^A \lambda v^{\forall x\, \neg A}.vxu. \qquad \square$$

Existence–Elimination–Lemma. $\vdash (\neg\neg B \to B) \to \exists x\, A \to (\forall x.A \to B) \to B$ *if* $x \notin \mathrm{FV}(B)$.

Proof. A derivation term is

$$\lambda u^{\neg\neg B \to B} \lambda v^{\neg\forall x \neg A} \lambda w^{\forall x.A \to B}.u\lambda u_2^{\neg B}.v\lambda x\lambda u_1^A.u_2(wxu_1). \qquad \square$$

Note that the stability assumption $\neg\neg B \to B$ is not needed if B does not contain an atom $\neq \bot$ as a strictly positive subformula. This will be the case for the derivations below, where B will always be an existential formula.

Let us now fix our language. We use a ternary relation symbol R to represent the graph of the function $y + 2^x$; so $R(y, x, z)$ is intended to mean $y + 2^x = z$. We now axiomatize R by means of Horn clauses. For simplicity we use a unary function symbol s (to be viewed as the successor function) and a constant 0; one could use logic without function symbols instead — as Orevkov does —, but this makes the formulas somewhat less readable and the proofs less perspicious.

$$\begin{aligned}
&\mathrm{Hyp}_1\colon \forall y\, R(y, 0, s(y))\\
&\mathrm{Hyp}_2\colon \forall y, x, z, z_1.R(y, x, z) \to R(z, x, z_1) \to R(y, s(x), z_1)
\end{aligned}$$

The goal formula then is

$$C_k := \exists z_k, \ldots, z_0.R(0, 0, z_k) \wedge R(0, z_k, z_{k-1}) \wedge \ldots \wedge R(0, z_1, z_0).$$

To obtain the short proof of the goal formula C_k we use formulas A_i with a free parameter x; for ease in reading we write $A[r]$ instead of $A[r/x]$.

$$\begin{aligned}
A_0 &:= \forall y \exists z\, R(y, x, z),\\
A_{i+1} &:= \forall y.A_i[y] \to \exists z.A_i[z] \wedge R(y, x, z).
\end{aligned}$$

Lemma. $\vdash \mathrm{Hyp}_1 \to \mathrm{Hyp}_2 \to A_i[0]$.

Proof. We give an informal argument, which can easily be converted into a formal proof. Note that the Existence–Elimination–Lemma is used only with existential formulas as conclusions. Hence it is not necessary to use stability axioms and we have a derivation in minimal logic.

Case $i = 0$. Obvious by Hyp_1.

Case $i = 1$. Let x with $A_0[x]$ be given. It is sufficient to show $A_0[s(x)]$, that is $\forall y \exists z_1\, R(y, s(x), z_1)$. So let y be given. We know

$$A_0[x] = \forall y \exists z\, R(y, x, z). \tag{1}$$

Applying (1) to our y gives z such that $R(y, x, z)$. Applying (1) again to this z gives z_1 such that $R(z, x, z_1)$. By Hyp_2 we obtain $R(y, s(x), z_1)$.

Case $i + 2$. Let x with $A_{i+1}[x]$ be given. It suffices to show $A_{i+1}[s(x)]$, that is $\forall y. A_i[y] \to \exists z. A_i[z] \wedge R(y, s(x), z)$. So let y with $A_i[y]$ be given. We know

$$A_{i+1}[x] = \forall y. A_i[y] \to \exists z_1. A_i[z_1] \wedge R(y, x, z_1). \tag{2}$$

Applying (2) to our y gives z such that $A_i[z]$ and $R(y, x, z)$. Applying (2) again to this z gives z_1 such that $A_i[z_1]$ and $R(z, x, z_1)$. By Hyp_2 we obtain $R(y, s(x), z_1)$. □

Remark. Note that the derivations given have a fixed length, independent of i.

Lemma. $\vdash \mathrm{Hyp}_1 \to \mathrm{Hyp}_2 \to C_k$.

Proof. We give an informal argument, which can easily be converted into a formal proof. Again the Existence–Elimination–Lemma is used only with existential formulas as conclusions, and hence we have a derivation in minimal logic.

$A_k[0]$ applied to 0 and $A_{k-1}[0]$ yields z_k with $A_{k-1}[z_k]$ and $R(0, 0, z_k)$.

$A_{k-1}[z_k]$ applied to 0 and $A_{k-2}[0]$ yields z_{k-1} with $A_{k-2}[z_{k-1}]$ and $R(0, z_k, z_{k-1})$.

$A_1[z_2]$ applied to 0 and $A_0[0]$ yields z_1 with $A_0[z_1]$ and $R(0, z_2, z_1)$.

$A_0[z_1]$ applied to 0 yields z_0 with $R(0, z_1, z_0)$. □

Remark. Note that the derivations given have length linear in k.

We want to compare the length of this derivation of C_k with the length of an arbitrary normal derivation.

Lemma. *Any normal derivation of C_k from* Hyp_1 *and* Hyp_2 *has at least 2_k nodes.*

Proof. Let a normal derivation d of falsity $\perp$ from Hyp_1, Hyp_2 and the additional hypothesis

$$u\colon \forall z_k, \dots, z_0. R(0, 0, z_k) \to R(0, z_k, z_{k-1}) \to \dots \to R(0, z_1, z_0) \to \perp$$

be given. We may assume that d does not contain free object variables (otherwise substitute them by 0). The main branch of d must begin with u, and its side premises are all of the form $R(0, s^n(0), s^k(0))$.

Observe that any normal derivation of $R(s^m(0), s^n(0), s^k(0))$ from Hyp_1, Hyp_2 and u has at least 2^n occurrences of Hyp_1 and is such that $k = m + 2^n$. This can be seen easily by induction on n. Note also that such a derivation cannot involve u.

If we apply this observation to the above derivations of the side premises we see that they derive

$$R(0, 0, s^{2^0}(0)), \quad R(0, s^{2^0}(0), s^{2^{2^0}}(0)), \quad \ldots \quad R(0, s^{2_{k-1}}(0), s^{2_k}(0)).$$

The last of these derivations uses at least $2^{2_{k-1}} = 2_k$–times Hyp_1. $\square$

5 The strong existential quantifier

We now extend our language $\mathcal{L}$ by a strong existential quantifier written $\exists^*$ (as opposed to $\exists$ defined by $\neg\forall\neg$). There are two approaches to deal with formulas containing $\exists^*$ in a constructive setting, e.g. in minimal or intuitionistic logic.

- A formula containing $\exists^*$ is considered not to be an entity the deduction system can deal with: some "realizing terms" are required to turn it into a "judgement" (this terminology is due to Weyl [34] and has been taken up by Martin–Löf). E.g. r realizes $\exists^* x\, A$ is a judgement, which can be translated into $A[r/x]$.
- (Heyting) The logic is extended by axioms expressing the intended meaning of the strong existential quantifier.

We will treat both approaches here. At first sight, Weyl's point of view is more convincing. However, Heyting's is more prominent in the literature, and we also need it to properly discuss Friedman's A–translation.

Let us first describe Heyting's approach. Here we extend our notion of an $\mathcal{L}$–formula by adding a clause

- If A is a formula, then $\exists^* x^\rho\, A$ is a formula.

In the inductive definition of derivation terms d^A in minimal logic and their sets $\mathrm{FA}(d^A)$ of free assumptions we have to add two more clauses:

($\exists^{*+}$) If A is a formula and x is a variable of type ρ, then

$$\exists^{*+}_{x,A} \colon \forall\forall x. A \to \exists^* x\, A$$

is a derivation term, where $\forall C$ denotes the universal closure of C, and $\mathrm{FA}(\exists^{*+}_{x,A}) = \emptyset$.

($\exists^{*-}$) If A, B are formulas and x is a variable of type ρ such that $x \notin \mathrm{FV}(B)$, then

$$\exists^{*-}_{x,A,B} \colon \forall.(\exists^* x^\rho\, A) \to (\forall x^\rho . A \to B) \to B$$

is a derivation term with $\mathrm{FA}(\exists^{*-}_{x,A,B}) = \emptyset$.

Clearly $\mathrm{FV}(\exists^{*+}) = \mathrm{FV}(\exists^{*-}) = \emptyset$.

For these new derivation terms we have the following conversion rule:

$$\exists^{*-}\vec{t}\left(\exists^{*+}\vec{t}t\, d^{A[t/x]}\right) e \mapsto etd. \qquad (\exists^*)$$

It can be shown that any derivation term has a unique normal form with respect to $\beta\eta\exists^*$–conversion.

An alternative (in fact more usual) way to introduce the strong existential quantifier $\exists^*$ into our natural deduction calculus is to use *rules* instead of axiom schemata. These rules have been formulated by Gentzen [11], as follows.

($\exists^{*+}$)

$$\dfrac{\begin{array}{c} | \\ A[t] \end{array}}{\exists^* x\, A}\,\exists^{*+}$$

($\exists^{*-}$)

$$\dfrac{\begin{array}{c} | \\ \exists^* x\, A \end{array} \qquad \begin{array}{c} [A] \\ | \\ B \end{array}}{B}\,\exists^{*-},$$

provided x does not occur free in B and in any open assumption of the given derivation of B apart from the assumption A shown.

It is easy to show that both calculi are equivalent.

Note that the calculus with $\exists^*$–rules can also be viewed as an inductive definition of a set of derivation terms, as in §2. We just have to add two clauses:

($\exists^{*+}$) If $d^{A[t]}$ is a derivation term, then

$$\exists^{*+}(t, d^{A[t]})^{\exists^* x\, A}$$

is a derivation term with $\mathrm{FA}(\exists^{*+}(t, d^{A[t]})) = \mathrm{FA}(d^{A[t)})$.

($\exists^{*-}$) If $d^{\exists^* x^\rho A}$ and e^B are derivation terms, u^A is an assumption variable and $x \notin \mathrm{FV}(B) \cup \bigcup\{\mathrm{FV}(C) : v^C \in \mathrm{FA}(e^B) \setminus \{u^A\}\}$, then

$$\exists^{*-}(d^{\exists^* x^\rho A}, x^\rho, u^A, e^B)^B$$

is a derivation term with $\mathrm{FA}(\exists^{*-}(d, x, u, e)) = \mathrm{FA}(d) \cup (\mathrm{FA}(e) \setminus \{u\})$. The variables x and u are viewed as bound by $\exists^{*-}(d, x, u, e)$.

Also the definition of the set $\mathrm{FV}(d)$ of free (object) variables has to be extended by

$$\begin{aligned} \mathrm{FV}(\exists^{*+}(t, d)) &:= \mathrm{FV}(d), \\ \mathrm{FV}(\exists^{*-}(d, x, u, e)) &:= \mathrm{FV}(d) \cup (\mathrm{FV}(e) \setminus \{x\}). \end{aligned}$$

Similarly one can also introduce a *strong disjunction* into our natural deduction calculus, written $\vee^*$ (as opposed to $\vee$ defined by $A \vee B := \neg(\neg A \wedge \neg B)$). We again extend our notion of an $\mathcal{L}$–formula by adding a clause

- If A, B are formulas, then $A \vee^* B$ is a formula.

In the inductive definition of derivation terms d^A in minimal logic and their sets $\mathrm{FA}(d^A)$ of free assumptions we have to add three more clauses:

($\vee_0^{*+}$) If A, B are formulas, then
$$\vee^{*+}_{A,B,0} \colon \forall . A \to A \vee^* B$$
is a derivation term, and $\mathrm{FA}(\vee^{*+}_{A,B,0}) = \emptyset$.

($\vee_1^{*+}$) If A, B are formulas, then
$$\vee^{*+}_{A,B,1} \colon \forall . B \to A \vee^* B$$
is a derivation term, and $\mathrm{FA}(\vee^{*+}_{A,B,1}) = \emptyset$.

($\vee^{*-}$) If A, B, C are formulas, then
$$\vee^{*-}_{A,B,C} \colon \forall . A \vee^* B \to (A \to C) \to (B \to C) \to C$$
is a derivation term with $\mathrm{FA}(\vee^{*-}_{A,B,C}) = \emptyset$.

Clearly $\mathrm{FV}(\vee_i^{*+}) = \mathrm{FV}(\vee^{*-}) = \emptyset$.

An alternative (and again more usual) way to introduce strong disjunction $\vee^*$ into our natural deduction calculus is to use *rules* instead of axiom schemata. These rules have been formulated by Gentzen [11], as follows.

($\vee_0^{*+}$)
$$\cfrac{\begin{array}{c}|\\ A\end{array}}{A \vee^* B}\,\vee_0^{*+}$$

($\vee_1^{*+}$)
$$\cfrac{\begin{array}{c}|\\ B\end{array}}{A \vee^* B}\,\vee_1^{*+}$$

($\vee^{*-}$)
$$\cfrac{\begin{array}{c}|\\ A \vee^* B\end{array} \qquad \begin{array}{c}[A]\\ |\\ C\end{array} \qquad \begin{array}{c}[B]\\ |\\ C\end{array}}{C}\,\vee^{*-}\ .$$

It is easy to show that both calculi are equivalent.

Note that the calculus with $\vee^*$–rules can again be viewed as an inductive definition of a set of derivation terms, as in §2. We just have to add three clauses:

($\vee_0^{*+}$) If d^A is a derivation term, then

$$\vee_0^{*+}(d^A)^{A\vee^* B}$$

is a derivation term with $\mathrm{FA}(\vee_0^{*+}(d^A)) = \mathrm{FA}(d^A)$.

($\vee_1^{*+}$) If e^B is a derivation term, then

$$\vee_1^{*+}(e^B)^{A\vee^* B}$$

is a derivation term with $\mathrm{FA}(\vee_1^{*+}(e^B)) = \mathrm{FA}(e^B)$.

($\vee^{*-}$) If $d^{A\vee^* B}$ and e^C, f^C are derivation terms and u^A, v^B are assumption variables, then

$$\vee^{*-}(d^{A\vee^* B}, u^A, e^C, v^B, f^C)^B$$

is a derivation term with $\mathrm{FA}(\vee^{*-}(d,u,e,v,f)) = \mathrm{FA}(d) \cup (\mathrm{FA}(e) \setminus \{u\}) \cup (\mathrm{FA}(f) \setminus \{v\})$. Again the variables u, v are viewed as bound by $\vee^{*-}(d,u,e,v,f)$.

Also the definition of the set $\mathrm{FV}(d)$ of free (object) variables has to be extended by

$$\begin{aligned}
\mathrm{FV}(\vee_0^{*+}(d^A)^{A\vee^* B}) &:= \mathrm{FV}(d) \cup \mathrm{FV}(B),\\
\mathrm{FV}(\vee_1^{*+}(e^B)^{A\vee^* B}) &:= \mathrm{FV}(e) \cup \mathrm{FV}(A),\\
\mathrm{FV}(\vee^{*-}(d,u,e,v,f)) &:= \mathrm{FV}(d) \cup \mathrm{FV}(e) \cup \mathrm{FV}(f).
\end{aligned}$$

Note that one can easily extend the Ex–falso–quodlibet Lemma to the present situation and prove $\bot \to A$ for an arbitrary formula A. In the cases $\exists^* x\, A$ and $A \vee^* B$ just use the corresponding introduction axiom (or rule).

As an application of normalization we obtain the so–called existence and disjunction properties of minimal and intuitionistic logic. To formulate it we introduce the notion of an *instance* of a formula possibly involving $\exists^*, \vee^*$. It is obtained by recursively replacing stricly positive subformulas $\exists^* x\, A$ by $A[t/x]$ and $A \vee^* B$ by A or B. More precisely, we give the following inductive definition.

Definition. *The relation "A' is an instance of A" is defined inductively by the following clauses.*

(i) A is an instance of A if A does not contain an $\exists^*$– or $\vee^*$–formula as strictly positive subformula.

(ii) *If B' is an instance of B, then $A \to B'$ is an instance of $A \to B$.*

(iii) *If A' is an instance of A and B' is an instance of B, then $A' \wedge B'$ is an instance of $A \wedge B$.*

(iv) *If A' is an instance of A, then $\forall x\, A'$ is an instance of $\forall x\, A$.*

(v) *If A' is an instance of A, then A' is an instance of $A \vee^* B$. If B' is an instance of B, then B' is an instance of $A \vee^* B$.*

(vi) *If A' is an instance of $A[t]$, then A' is an instance of $\exists^* x\, A$.*

Note that if A' is an instance of A, then $\vdash A' \to A$.

Theorem. *Let $d[u_1\colon A_1, \ldots, u_n\colon A_n]\colon B$ where $A_1, \ldots, A_n$ do not contain an $\exists^*$– or $\vee^*$–formula as strictly positive subformula. Then we can find an instance B' of B and a derivation $d'[u_1\colon A_1, \ldots, u_n\colon A_n]\colon B'$.*

Proof by an analysis of the normal form of d. □

We can also show that the logic with $\exists^*$ and $\vee^*$ is conservative over the fragment without these connectives. This can either be proved using the modified realizabi;ity interpretation to be introduced in the next section, or else directly by an analysis of the normal form of the derivation.

6 Realizing terms

Let us now describe Weyl's approach. We restrict ourselves to formulas without $\vee^*$, since in the presence of a ground type of booleans we can define $A \vee^* B$ by

$$A \vee^* B := \exists^* p.(p = \mathrm{true} \to A) \wedge (p = \mathrm{false} \to B).$$

We define *judgements* to be expressions of the form

$$r_1^{\rho_1}, \ldots, r_m^{\rho_m} \,\mathrm{mr}\, A$$

(to be read $r_1^{\rho_1}, \ldots, r_m^{\rho_m}$ modified realizes A), where A is a formula built from atomic formulas using $\to$, $\wedge$, $\forall$ and $\exists^*$, and $\rho_1, \ldots, \rho_m = \tau(A)$ is a list of types associated with A, defined as follows.

$$\tau(R(\vec{t})) := \varepsilon,$$

where ε denotes the empty list, and if $\tau(A) = \vec{\rho}$ and $\tau(B) = \sigma_1, \ldots, \sigma_n$ we let

$$\begin{aligned}
\tau(A \to B) &:= \vec{\rho} \to \sigma_1, \ldots, \vec{\rho} \to \sigma_n, \\
\tau(A \wedge B) &:= \vec{\rho}, \vec{\sigma}, \\
\tau(\forall x^\rho\, B) &:= \rho \to \sigma_1, \ldots, \rho \to \sigma_n, \\
\tau(\exists^* x^\rho\, B) &:= \rho, \vec{\sigma}.
\end{aligned}$$

Instead of $\vec{\rho} \to \sigma_1, \ldots, \vec{\rho} \to \sigma_n$ we will sometimes write $\vec{\rho} \to \vec{\sigma}$. To give some examples, let n, m, k be of type nat. Then

$$\begin{aligned}
\tau(\forall n \exists^* m\, R(n,m)) &= \mathrm{nat} \to \mathrm{nat}, \\
\tau(\forall n \exists^* m \exists^* k\, R(n,m,k) &= (\mathrm{nat} \to \mathrm{nat}), (\mathrm{nat} \to \mathrm{nat}), \\
\tau(\forall n \exists^* m\, R(n,m) \to \exists^* k\, Q(k)) &:= (\mathrm{nat} \to \mathrm{nat}) \to \mathrm{nat}.
\end{aligned}$$

Note that $\tau(A) = \varepsilon$ iff A is a *Harrop formula* (i.e. contains $\exists^*$ in premises of $\to$ only).

For any judgement we now define its *modified realizability interpretation*, i.e. its translation in our $\exists^*$–free language.

$$\begin{aligned}
\varepsilon \,\mathrm{mr}\, R(\vec{t}) &:= R(\vec{t}), \\
r_1, \ldots, r_n \,\mathrm{mr}\, (A \to B) &:= \forall \vec{x}.\vec{x} \,\mathrm{mr}\, A \to r_1\vec{x}, \ldots, r_n\vec{x} \,\mathrm{mr}\, B, \\
\vec{r}, \vec{s} \,\mathrm{mr}\, (A \wedge B) &:= \vec{r} \,\mathrm{mr}\, A \wedge \vec{s} \,\mathrm{mr}\, B, \\
r_1, \ldots, r_n \,\mathrm{mr}\, \forall x^\rho\, B &:= \forall x^\rho\, r_1 x, \ldots, r_n x \,\mathrm{mr}\, B, \\
r, \vec{s} \,\mathrm{mr}\, \exists^* x^\rho\, B &:= \vec{s} \,\mathrm{mr}\, B[r/x].
\end{aligned}$$

Note that for Harrop formulas A we have $\varepsilon \,\mathrm{mr}\, A \equiv A$ iff A does not contain $\exists^*$.

Let us now set up a relation between the implicit and the explicit approach to deal with the existential quantifier.

Definition. *Assume that to any assumption variable u^B we have assigned a list $\vec{x}_u^{\tau(B)} = x_{u1}^{\rho_1}, \dots, x_{un}^{\rho_n}$ of distinct variables, where $\rho_1, \dots, \rho_n = \tau(B)$. Relative to this assignment we define for any derivation d^A its extracted terms $\mathrm{ets}(d^A)$, by induction on d^A. If $\tau(A) = \sigma_1, \dots, \sigma_k$, then $\mathrm{ets}(d^A)$ will be a list $r_1^{\sigma_1}, \dots, r_k^{\sigma_k}$.*

$$
\begin{aligned}
\mathrm{ets}(u^A) &= \vec{x}_u^{\tau(A)}, \\
\mathrm{ets}(\lambda u^A\, d^B) &= \lambda \vec{x}_u^{\tau(A)}\, \mathrm{ets}(d), \\
\mathrm{ets}(d^{A\to B} e^A) &= \mathrm{ets}(d)\mathrm{ets}(e), \\
\mathrm{ets}(\langle d^A, e^B \rangle) &= \mathrm{ets}(d^A), \mathrm{ets}(e^B), \\
\mathrm{ets}(\pi_0(d^{A\wedge B})) &= \textit{the head of } \mathrm{ets}(d^{A\wedge B}) \textit{ of same length as } \tau(A), \\
\mathrm{ets}(\pi_1(d^{A\wedge B})) &= \textit{the tail of } \mathrm{ets}(d^{A\wedge B}) \textit{ of same length as } \tau(B), \\
\mathrm{ets}(\lambda x^\rho\, d^A) &= \lambda x^\rho\, \mathrm{ets}(d), \\
\mathrm{ets}(d^{\forall x^\rho\, A} t^\rho) &= \mathrm{ets}(d)t, \\
\mathrm{ets}(\exists^{*+}_{x,A}) &= \lambda \vec{x} \lambda x \lambda \vec{y}. x, \vec{y}, \\
\mathrm{ets}(\exists^{*-}_{x,A,B}) &= \lambda \vec{x} \lambda x \lambda \vec{y} \lambda z_1 \dots \lambda z_n . z_1 x \vec{y}, \dots, z_n x \vec{y}.
\end{aligned}
$$

Note that if $\mathrm{ets}(d) = r_1, \dots, r_k$ and $\mathrm{ets}(e) = \vec{s}$, then $\mathrm{ets}(d)\mathrm{ets}(e) = r_1\vec{s}, \dots, r_k\vec{s}$ and $\lambda\vec{x}\, \mathrm{ets}(d) = \lambda\vec{x}\, r_1, \dots, \lambda\vec{x}\, r_k$. In the last clause the (omitted) types are

$$
x^\rho, \quad (\vec{y})^{\tau(A)} \quad \text{and} \quad z_j^{\rho\to\tau(A)\to\sigma_j},
$$

where $\tau(B) = \sigma_1, \dots, \sigma_n$.

The following can be proved easily.

Lemma. $\mathrm{FV}(\mathrm{ets}(d)) \subseteq \mathrm{FV}(d) \cup \{\vec{x}_u^{\tau(A)} : u^A \in \mathrm{FA}(d)\}$.

Lemma. *We have* $\mathrm{ets}(d[t/x]) = \mathrm{ets}(d)[t/x]$ *and* $\mathrm{ets}(d[e/u]) = \mathrm{ets}(d)[\mathrm{ets}(e)/\vec{x}_u]$, *hence*

$$
\begin{aligned}
d =_{\beta\exists^*} e &\implies \mathrm{ets}(d) =_\beta \mathrm{ets}(e) \\
d =_\eta e &\implies \mathrm{ets}(d) =_\eta \mathrm{ets}(e)
\end{aligned}
$$

□

Hence we can safely identify terms with the same $\beta\eta\exists^*$–normal forms.

Soundness Theorem. *Assume that to any assumption variable u^A we have assigned a list $\vec{x}_u^{\tau(A)}$ and a new assumption variable $\tilde{u}\colon \vec{x}_u^{\tau(A)}\, \mathrm{mr}\, A$. Relative to this assignment we can find for any derivation $d\colon A$ with $\mathrm{FA}(d) = \{u_1\colon A_1, \dots, u_n\colon A_n\}$ a derivation*

$$
\mu(d)\colon \mathrm{ets}(d)\, \mathrm{mr}\, A
$$

with $\mathrm{FA}(\mu(d)) = \{\tilde{u}_1\colon \vec{x}_1\, \mathrm{mr}\, A_1, \dots, \tilde{u}_n\colon \vec{x}_n\, \mathrm{mr}\, A_n\}$.

Proof by induction on d.

Case $u_i: A_i$. Then $\tilde{u}_i: \vec{x}_{u_i} \,\mathrm{mr}\, A_i$.

Case $\lambda u^B\, d^A$. Let $\tau(B) = \vec{\rho}$ and $\tau(A) = \sigma_1, \ldots, \sigma_m$. Then we have $\tau(B \to A) = \vec{\rho} \to \sigma_1, \ldots, \vec{\rho} \to \sigma_m$. We look for a derivation of

$$\lambda \vec{x}^{\vec{\rho}}\, \mathrm{ets}(d^A) \,\mathrm{mr}\, (B \to A),$$

i.e. of

$$\forall \vec{x}^{\vec{\rho}}.\vec{x} \,\mathrm{mr}\, B \to \mathrm{ets}(d^A) \,\mathrm{mr}\, A,$$

since $(\lambda \vec{x}^{\vec{\rho}}\, \mathrm{ets}(d^A))\vec{x} =_\beta \mathrm{ets}(d^A)$ and terms with the same β–normal form are identified. Hence we can take

$$\mu(\lambda u^B\, d^A) := \lambda \vec{x} \lambda \tilde{u}^{\vec{x} \,\mathrm{mr}\, B} \mu(d).$$

Case $d^{A \to B} e^A$. By IH we have

$$\begin{aligned} &\mu(d): \mathrm{ets}(d) \,\mathrm{mr}\, (A \to B), \\ &\mu(e): \mathrm{ets}(e) \,\mathrm{mr}\, A. \end{aligned}$$

By definition the first of these means

$$\mu(d): \forall \vec{x}.\vec{x} \,\mathrm{mr}\, A \to \mathrm{ets}(d)\vec{x} \,\mathrm{mr}\, B.$$

Hence we have

$$\mu(d)\mathrm{ets}(e)\mu(e): \mathrm{ets}(d)\mathrm{ets}(e) \,\mathrm{mr}\, B$$

and since $\mathrm{ets}(de) = \mathrm{ets}(d)\mathrm{ets}(e)$ we can take $\mu(de) := \mu(d)\mathrm{ets}(e)\mu(e)$.

Case $\langle d^A, e^B \rangle$. By IH we have

$$\begin{aligned} &\mu(d): \mathrm{ets}(d) \,\mathrm{mr}\, A, \\ &\mu(e): \mathrm{ets}(e) \,\mathrm{mr}\, B, \end{aligned}$$

hence

$$\langle \mu(d), \mu(e) \rangle: \mathrm{ets}(d) \,\mathrm{mr}\, A \wedge \mathrm{ets}(e) \,\mathrm{mr}\, B.$$

Now since $\mathrm{ets}(d) \,\mathrm{mr}\, A \wedge \mathrm{ets}(e) \,\mathrm{mr}\, B \equiv \mathrm{ets}(d), \mathrm{ets}(e) \,\mathrm{mr}\, (A \wedge B) \equiv \mathrm{ets}(\langle d, e \rangle) \,\mathrm{mr}\, (A \wedge B)$ it suffices to define $\mu(\langle d, e \rangle) := \langle \mu(d), \mu(e) \rangle$.

Case $\pi_0(d^{A \wedge B})$. By IH we have

$$\mu(d): \mathrm{ets}(d) \,\mathrm{mr}\, (A \wedge B).$$

By the definition of modified realizability $\mu(d)$ proves a conjunction. Let ℓ be the length of $\tau(A)$ and $\mathrm{head}_\ell(\mathrm{ets}(d))$ the head of $\mathrm{ets}(d)$ of length ℓ. Then

$$\pi_0(\mu(d)): \mathrm{head}_\ell(\mathrm{ets}(d)) \,\mathrm{mr}\, A.$$

So it suffices to define $\mu(\pi_0(d)) := \pi_0(\mu(d))$.

Case $\pi_1(d^{A\wedge B})$. Similar.

Case $\lambda x^\rho\, d^A$. We have to find a derivation of

$$\lambda x^\rho \mathrm{ets}(d)\,\mathrm{mr}\,\forall x^\rho A,$$

i.e. of (since we identify terms with the same β–normal forms)

$$\forall x^\rho\, \mathrm{ets}(d)\,\mathrm{mr}\, A.$$

By IH we have $\mu(d)\colon \mathrm{ets}(d)\,\mathrm{mr}\, A$ with free assumptions not involving x. Hence we can take $\mu(\lambda x^\rho\, d) := \lambda x^\rho\, \mu(d)$.

Case $d^{\forall x^\rho\, A} t^\rho$. By IH we have

$$\mu(d)\colon \mathrm{ets}(d)\,\mathrm{mr}\,\forall x\, A,$$

i.e.

$$\mu(d)\colon \forall x\, \mathrm{ets}(d)x\,\mathrm{mr}\, A.$$

Hence $\mu(d)t\colon \mathrm{ets}(d)t\,\mathrm{mr}\, A[t/x]$ and we can take $\mu(dt) := \mu(d)t$.

Case $\exists^{*+}_{x,A}$. We look for a derivation $\mu(\exists^{*+}_{x,A})$ of

$$\begin{gathered}
(\lambda\vec{x}\lambda x\lambda\vec{y}.x,\vec{y})\,\mathrm{mr}\,\forall\vec{x}\forall x^\rho.A \to \exists^* x^\rho\, A\\
\forall\vec{x}\forall x^\rho(\lambda\vec{y}.x,\vec{y})\,\mathrm{mr}\,(A \to \exists^* x^\rho\, A)\\
\forall\vec{x}\forall x^\rho\forall\vec{y}.\vec{y}\,\mathrm{mr}\, A \to x,\vec{y}\,\mathrm{mr}\,\exists^* x^\rho\, A\\
\forall\vec{x}\forall x^\rho\forall\vec{y}.\vec{y}\,\mathrm{mr}\, A \to \vec{y}\,\mathrm{mr}\, A
\end{gathered}$$

Hence we can take $\mu(\exists^{*+}_{x,A}) := \lambda\vec{x}\lambda x\lambda\vec{y}\lambda u\, u$ where $u\colon \vec{y}\,\mathrm{mr}\, A$.

Case $\exists^{*-}_{x,A,B}$. Recall that

$$\exists^{*-}_{x,A,B}\colon \forall\vec{x}.\exists^* x\, A \to (\forall x.A \to B) \to B$$

with $x \notin \mathrm{FV}(B)$. We look for a derivation $\mu(\exists^{*-}_{x,A,B})$ of

$$\begin{gathered}
(\lambda\vec{x}\lambda x\lambda\vec{y}\lambda z_1\ldots\lambda z_n.z_1 x\vec{y},\ldots,z_n x\vec{y})\,\mathrm{mr}\,\forall\vec{x}.\exists^* x\, A \to (\forall x.A \to B) \to B\\
\forall\vec{x}(\lambda x\lambda\vec{y}\lambda z_1\ldots\lambda z_n.z_1 x\vec{y},\ldots,z_n x\vec{y})\,\mathrm{mr}\,(\exists^* x\, A \to (\forall x.A \to B) \to B)\\
\forall\vec{x}\forall x\forall\vec{y}.x,\vec{y}\,\mathrm{mr}\,\exists^* x\, A \to (\lambda z_1\ldots\lambda z_n.z_1 x\vec{y},\ldots,z_n x\vec{y})\,\mathrm{mr}\,((\forall x.A \to B) \to B)\\
\forall\vec{x}\forall x\forall\vec{y}.\vec{y}\,\mathrm{mr}\, A \to \forall\vec{z}.\vec{z}\,\mathrm{mr}\,(\forall x.A \to B) \to z_1 x\vec{y},\ldots,z_n x\vec{y}\,\mathrm{mr}\, B
\end{gathered}$$

Now since modulo β–equivalence $\vec{z}\,\mathrm{mr}\,\forall x.A \to B$ is the same as

$$\forall x\forall\vec{y}.\vec{y}\,\mathrm{mr}\, A \to z_1 x\vec{y},\ldots,z_n x\vec{y}\,\mathrm{mr}\, B$$

we can easily derive this formula by

$$\mu(\exists^{*-}_{x,A,B}) := \lambda\vec{x}\lambda x\lambda\vec{y}\lambda u\lambda\vec{z}\lambda v.vx\vec{y}u,$$

where $u\colon \vec{y}\,\mathrm{mr}\, A$ and $v\colon \forall x\forall\vec{y}.\vec{y}\,\mathrm{mr}\, A \to z_1 x\vec{y},\ldots,z_n x\vec{y}\,\mathrm{mr}\, B$. □

Remark: Mints has shown already in [14] that for any $d\colon \exists^* x\, A$ with $\mathrm{FA}(d) = \mathrm{FV}(d) = \emptyset$ the extracted terms $\mathrm{ets}(d)$ reduce into the terms that can be read off from the long normal form of $d\colon \exists^* x\, A$. This has later been generalized by Stein in his thesis [28].

7 Arithmetic

Let us now extend these considerations to arithmetic. We allow constants for primitive recursive functionals of arbitrary types (i.e. terms of Gödel's T), identifying terms with the same normal form (w.r.t. the usual conversion rules for Gödel's T). It is assumed that at least the ground types nat of natural numbers and boole of booleans are present. We restrict ourselves to decidable atomic formulas; it is convenient to represent them by boolean terms, i.e. in the form $\mathrm{atom}(t^{\mathrm{boole}})$ where atom is a distinguished relation symbol. We could equally well take equations $r = s$ with r, s terms of type nat as the only boolean formulas. We do not need $\bot$ as an extra atomic formula, since it can be defined by $\bot := \mathrm{atom}(\mathrm{false})$. Let us use n, m as variables of type nat and p, q as variables of type boole. Our *induction schemata* are the universal closures of

$$A[0/n] \to (\forall n.A \to A[n+1/n]) \to \forall n\, A,$$
$$A[\mathrm{true}/p] \to A[\mathrm{false}/p] \to \forall p\, A.$$

We can use boolean induction (i.e. case analysis) to prove stability $\neg\neg\mathrm{atom}(p) \to \mathrm{atom}(p)$ and from this we can as before conclude the stability $\neg\neg A \to A$ of formulas A built from atoms by $\to$, $\wedge$ and $\forall$. As already remarked, strong disjunction $\vee^*$ can be defined by means of the strong existential quantifier $\exists^*$.

Let us now carry out this program. First we extend the notion of a term by adding the clauses

- The constants 0^{nat}, $S^{\mathrm{nat}\to\mathrm{nat}}$, $\mathrm{true}^{\mathrm{boole}}$ and $\mathrm{false}^{\mathrm{boole}}$ are terms.
- For each type ρ the constants

  $$R_{\mathrm{nat},\rho}\colon \rho \to (\mathrm{nat} \to \rho \to \rho) \to \mathrm{nat} \to \rho,$$
  $$R_{\mathrm{boole},\rho}\colon \rho \to \rho \to \mathrm{boole} \to \rho$$

  are terms.

We add the following conversion rules (writing $t+1$ for St).

$$R_{\mathrm{nat},\rho} rs0 \to_0 r,$$
$$R_{\mathrm{nat},\rho} rs(t+1) \to_0 st(R_{\mathrm{nat},\rho} rst)$$

and

$$R_{\mathrm{boole},\rho} rs\, \mathrm{true} \to_0 r,$$
$$R_{\mathrm{boole},\rho} rs\, \mathrm{false} \to_0 s.$$

We now show that addition of these conversion rules for R does not destroy the property of our reduction relation to be terminating.

There are many treatments of this problem in the literature. Troelstra in [30], p. 107–108 gives a proof which either uses König's Lemma (or the fan theorem) or else the Church–Rosser Theorem for $\to$. A proof which works for closed terms only (but in a general setting where bounded fixed point operators are allowed) is given in [26]. The short proof below is due to Ulrich Berger; it works for arbitrary arithmetical terms and does not use the Church–Rosser Theorem.

Theorem. $\to$ *is terminating.*

Proof. We extend the argument in §4, which was based on Tait's strong computability predicates. Let us write $\mathrm{SC}(r)$ to mean that r is strongly computable, and $\mathrm{SN}(r)$ to mean that r is strongly normalizeable. It remains to be shown that the constants $R_{\mathrm{nat},\rho}$ and $R_{\mathrm{boole},\rho}$ are strongly computable. We restrict ourselves to $R_{\mathrm{nat},\rho}$; for $R_{\mathrm{boole},\rho}$ the argument is similar (and simpler). So we have to show that for any terms $r, s, t, \vec{t}$

$$\mathrm{SC}(r, s, t, \vec{t}) \implies \mathrm{SN}(Rrst\vec{t}), \tag{1}$$

since for ground types SC and SN coincide. So assume $\mathrm{SC}(r, s, t, \vec{t})$. We prove (1) by induction on triples

$$(h(r, s, t), \#t, h(\vec{t}))$$

ordered lexicographically, where $\#r$ is the length of r and $h(r_1, \dots, r_n)$ is the sum of the heights of the reductions trees for $r_1, \dots, r_n$ (assuming $\mathrm{SN}(r_1, \dots, r_n)$); here we have to use König's Lemma. It clearly suffices to show

$$\forall t'. Rrst\vec{t} \to t' \implies \mathrm{SN}(t').$$

If the reduction takes place within a subterm $r, s, t, \vec{t}$ the assigned triple gets smaller and the claim follows by IH (here we need that strong computability is preserved under conversion steps; cf. Lemma 2 in §4). Since the case $t \equiv 0$, $t' \equiv r\vec{t}$ is trivial it suffices to consider the case $t \equiv t_0 + 1$, $t' \equiv st_0(Rrst_0)\vec{t}$. For all $\vec{u}$ with $\mathrm{SC}(\vec{u})$ we have

$$(h(r, s, t_0), \#t_0, h(\vec{u})) < (h(r, s, t), \#t, h(\vec{t})).$$

Hence by IH for any terms $\vec{u}$

$$\mathrm{SC}(\vec{u}) \implies \mathrm{SN}(Rrst_0\vec{u}),$$

since clearly $\mathrm{SN}(t_0)$. Therefore $\mathrm{SC}(Rrst_0)$ and we obtain $\mathrm{SN}(t')$. $\square$

Now as in §4 we can conclude via Newman's Lemma that the normal form is uniquely determined. For simplicity we identify terms with the same $\beta\eta R$–normal form. Hence any closed term of type nat is identified with a term of the form $S(S(S \dots (S0) \dots))$ and any closed term of type boole is identified with either true or false. Such terms are denoted by $\underline{n}$ and called *numbers* (even if they are of type boole).

Let us consider some *examples* of arithmetical terms. Addition $n + m$ can be defined easily by recursion on m; note that the parameter n remains fixed, since $n + (Sm) = S(n + m)$. Let

$$+ := \lambda n, m.R_{\mathrm{nat},nat} n(\lambda x, y.Sy)m.$$

Then we have

$$\begin{aligned} +n0 &\to Rn(\lambda x, y.Sy)0 \to n, \\ +n(Sm) &\to Rn(\lambda x, y.Sy)(Sm) \\ &\to (\lambda x, y.Sy)m(Rn(\lambda x, y.Sy)m) \\ &\to S(Rn(\lambda x, y.Sy)m) \\ &=_\beta S(+nm). \end{aligned}$$

Similarly one can define the predecessor pred and the zero-test zero?: nat $\to$ boole. Equality $n = m$ presents a slight problem, since in a definition by recursion on n the parameter m has to be changed: we must define $Sn = Sm$ by $n = m$. Therefore we represent $=$: nat $\to$ nat $\to$ boole as the function which maps n to $\lambda m.n = m$. More formally,

$$\mathrm{equal?} := \lambda n, m.R_{\mathrm{nat},\mathrm{nat}\to\mathrm{boole}} fgnm$$

with

$$\begin{aligned} f &:= \mathrm{zero?}, \\ g &:= \lambda n, h, m.R_{\mathrm{boole},\mathrm{boole}}\, \mathrm{false}\, (h(\mathrm{pred}\, m))(\mathrm{zero?}\, m). \end{aligned}$$

Then we have

$$\begin{aligned} \mathrm{equal?}\, 00 &\to Rfg00 \to \mathrm{zero?}\, 0 \to \mathrm{true}, \\ \mathrm{equal?}\, 0(Sm) &\to Rfg0(Sm) \to \mathrm{zero?}(Sm) \to \mathrm{false}, \\ \mathrm{equal?}\, (Sn)0 &\to Rfg(Sn)0 \to gn(Rfgn)0 \to \mathrm{false}, \\ \mathrm{equal?}\, (Sn)(Sm) &\to Rfg(Sn)(Sm) \\ &\to gn(Rfgn)(Sm) \\ &\to R_{\mathrm{boole},\mathrm{boole}}\, \mathrm{false}\, (Rfgnm)(\mathrm{zero?}\, (Sm)) \\ &\to Rfgnm \\ &=_\beta \mathrm{equal?}\, nm. \end{aligned}$$

Problem: Define a closed term f: nat $\to$ nat representing the Fibonacci sequence

$$\begin{aligned} &f0 = 1, \\ &f1 = 1, \\ &f(n + 2) = fn + f(n + 1). \end{aligned}$$

Hint: Define an auxiliary term $\overline{f}\colon \mathrm{nat} \to \mathrm{nat} \to \mathrm{nat}$ such that

$$\overline{f}nm = \begin{cases} fm, & \text{if } m < n, \\ 0 & \text{otherwise.} \end{cases}$$

Formulas are built from atomic formulas by means of $\to$, $\wedge$, $\forall$ and $\exists^*$. We let $\bot :=$ atom(false), $\neg A := A \to \bot$ and (writing p for atom(p)) $A \vee^* B := \exists^* p.(p \to A) \wedge (\neg p \to B)$.

We also extend the notion of a derivation term by constants for the truth axiom and induction axioms. Hence derivation terms in arithmetic are obtained by adding the clauses

(T) $\mathrm{T}\colon \mathrm{atom}(\mathrm{true})$ is a derivation term with $\mathrm{FA}(\mathrm{T}) = \emptyset$.

(Ind) For any formula $\forall n\, A$

$$\mathrm{Ind}_{n,A}\colon \forall.A[0/n] \to (\forall n.A \to A[n+1/n]) \to \forall n\, A$$

is a derivation term with $\mathrm{FA}(\mathrm{Ind}_{n,A}) = \emptyset$. Similarly, for any formula $\forall p\, A$

$$\mathrm{Ind}_{p,A}\colon \forall.A[\mathrm{true}/p] \to A[\mathrm{false}/p] \to \forall p\, A$$

is a derivation term with $\mathrm{FA}(\mathrm{Ind}_{p,A}) = \emptyset$.

Clearly $\mathrm{FV}(\mathrm{T}) := FV(\mathrm{Ind}_{n,A}) := \mathrm{FV}(\mathrm{Ind}_{p,A}) = \emptyset$.

In the sequel $\vdash$ refers to derivability in the arithmetical system determined by the derivation terms just described.

$\mathrm{Stab}_{\mathrm{atom}}\colon \forall p.\neg\neg\mathrm{atom}(p) \to \mathrm{atom}(p)$ can now be proved easily by boolean induction, using the truth axiom in the case true. As in §2 we can conclude $\vdash \neg\neg A \to A$ for formulas A without $\exists^*$, i.e. built with $\to$, $\wedge$ and $\forall$. We also have $\vdash \bot \to A$ for any A. Furthermore we can derive the usual axioms for $\vee^*$.

Problem: Give a derivation of $\forall n\, \mathrm{atom}(\mathrm{equal?}nn)$ for the closed term $\mathrm{equal?}\colon \mathrm{nat} \to \mathrm{nat} \to \mathrm{boole}$ defined above.

For these new derivation terms we have the following conversion rules.

$$\begin{aligned} \mathrm{Ind}_{n,A}\vec{r}de0 &\to_0 d, \\ \mathrm{Ind}_{n,A}\vec{r}de(t+1) &\to_0 et(\mathrm{Ind}_{n,A}\vec{r}det) \end{aligned}$$

and

$$\begin{aligned} \mathrm{Ind}_{p,A}\vec{r}de\,\mathrm{true} &\to_0 d, \\ \mathrm{Ind}_{p,A}\vec{r}de\,\mathrm{false} &\to_0 e. \end{aligned}$$

Again it can be shown by standard methods — just as for Gödel's T, cf. [30] — that any derivation term in arithmetic has a unique $\beta\eta\exists^* R$–normal form, where R refers to the conversion rules above.

The notion of extracted terms can straightforwardly be extended to this situation, and the Soundness Theorem carries over easily. In the case of $\mathrm{Ind}_{n,A}$ we have to prove

$$\mathrm{ets}(\mathrm{Ind}_{n,A})\,\mathrm{mr}\,\forall\vec{x}.A[0/n] \to (\forall n.A \to A[n+1/n]) \to \forall n\, A,$$

i.e.

$$\begin{aligned}\forall\vec{x}\forall\vec{y}\forall\vec{f}\forall n.\vec{y}\,\mathrm{mr}\,A[0/n] \to (\forall n\forall\vec{y}_1.\vec{y}_1\,\mathrm{mr}\,A \to \vec{f}n\vec{y}_1\,\mathrm{mr}\,A[n+1/n])&\\ \to \mathrm{ets}(\mathrm{Ind}_{n,A})\vec{x}\vec{y}\vec{f}n\,\mathrm{mr}\,A.&\end{aligned}$$

Hence we let $\mathrm{ets}(\mathrm{Ind}_{n,A}) := \lambda\vec{x}.R_1,\ldots,R_k$ where k is the length of $\tau(A) = \rho_1,\ldots,\rho_k$ and $R_1,\ldots,R_k$ are simultaneous primitive recursion operators of type $R_i\colon\vec{\rho} \to (\mathrm{nat} \to \vec{\rho} \to \vec{\rho}) \to \mathrm{nat} \to \rho_i$ satisfying

$$\begin{aligned}R_i\vec{y}\vec{f}0 &= y_i,\\ R_i\vec{y}\vec{f}(z+1) &= f_i z(R_1\vec{y}\vec{f}z)\ldots(R_k\vec{y}\vec{f}z);\end{aligned}$$

here $=$ denotes equality of $\beta\eta R$–normal forms. Using these equations we can then prove the above claim easily (recall that terms with the same normal form are identified). The operators $R_1,\ldots,R_n$ can be defined easily from the recursion constant $R_{\mathrm{nat},\rho_1\times\ldots\times\rho_k}$. We could equally well have introduced them as constants and added the equations above as conversion rules.

Boolean induction, i.e. case analysis is treated similarly. We let

$$\mathrm{ets}(\mathrm{Ind}_{p,A}) := \lambda\vec{x}.R_1,\ldots,R_k,$$

where now $R_1,\ldots,R_k$ are simultaneous primitive recursion (or case splitting) operators of type $R_i\colon\vec{\rho} \to \vec{\rho} \to \mathrm{boole} \to \rho_i$ satisfying

$$\begin{aligned}R_i\vec{y}\vec{z}\,\mathrm{true} &= y_i,\\ R_i\vec{y}\vec{z}\,\mathrm{false} &= z_i.\end{aligned}$$

The lemmata in §6 stating that ets commutes with substitution and reduction remain valid since the conversion rules for induction and recursion fit together.

The following remarks and definitions will be helpful later. Let us call a formula A *decidable* if there is a term t_A such that $\vdash A \leftrightarrow \mathrm{atom}(t_A)$.

1. Every quantifier–free formula is decidable. First let $\supset := \lambda p\lambda q.Rq\,\mathrm{true}\,p$ and $\& := \lambda p\lambda q.R(R\,\mathrm{true}\,\mathrm{false}\,q)\,\mathrm{false}\,p$. Clearly

$$\begin{aligned}&\forall p,q.(\mathrm{atom}(p) \to \mathrm{atom}(q)) \leftrightarrow \mathrm{atom}(\supset pq)\\ &\forall p,q.(\mathrm{atom}(p) \wedge \mathrm{atom}(q)) \leftrightarrow \mathrm{atom}(\& pq)\end{aligned}$$

are provable. Hence we let

$$\begin{aligned} t_{\mathrm{atom}(r)} &:= r, \\ t_{A\to B} &:= \supset t_A t_B \\ t_{A\wedge B} &:= \& t_A t_B. \end{aligned}$$

2. We can do case splitting according to decidable formulas A, i.e. for every formula $B[\vec{x}]$ we can prove

$$\mathrm{Cases}_{A,B}\colon (A \to B) \to (\neg A \to B) \to B.$$

The derivation $\mathrm{Cases}_{A,B}$ is given by

$$\lambda u_1, u_2.\mathrm{Ind}\,\vec{x}(\lambda u_3\lambda u_4.u_3\mathrm{T})(\lambda u_5\lambda u_6.u_6\neg\mathrm{F})t_A(\lambda u_7.u_1(d_1u_7))(\lambda u_8.u_2(d_2u_8))$$

where $d_1^{\mathrm{atom}(t_A)\to A}$, and $d_2^{\neg\mathrm{atom}(t_A)\to\neg A}$ are derivations which exist according to 1 and the axioms and assumption variables with indices are (writing t for $\mathrm{atom}(t)$)

$$\mathrm{Ind}_{p,(p\to B)\to(\neg p\to B)\to B},$$

$u_1^{A\to B}$, $u_2^{\neg A\to B}$, $u_3^{\mathrm{true}\to B}$, $u_4^{\neg\mathrm{true}\to B}$, $u_5^{\mathrm{false}\to B}$, $u_6^{\neg\mathrm{false}\to B}$, $u_7^{t_A}$ and $u_8^{\neg t_A}$. The extracted terms of $\mathrm{Cases}_{A,B}$ are given by

$$\mathrm{ets}(\mathrm{Cases}_{A,B}) = \lambda\vec{y},\vec{z}.\mathbf{if}\,t_A\vec{y}\vec{z} =_\eta \mathbf{if}\,t_A$$

where $\mathbf{if} := \vec{R}(\lambda\vec{y}_1,\vec{z}_1.\vec{y}_1)(\lambda\vec{y}_1,\vec{z}_1.\vec{z}_1)$ and $\vec{y},\vec{z},\vec{y}_1,\vec{z}_1$ are lists of variables of types $\vec{\rho} := \tau(B)$. Clearly

$$\mathbf{if}\,\mathrm{true}\,\vec{r}\vec{s} =_{\beta R} \vec{r}, \quad \mathbf{if}\,\mathrm{false}\,\vec{r}\vec{s} =_{\beta R} \vec{s}.$$

For better readability we use for $\mathbf{if}\,t_A\vec{r}\vec{s}$ the notation

$$\mathbf{if}\quad A\quad \mathbf{then}\quad \vec{r}\quad \mathbf{else}\quad \vec{s}\quad \mathbf{fi}.$$

Up to now we have restricted ourselves to the "neutral" system of arithmetic without extensionality axioms (this system is denoted by $N\text{–}HA^\omega$ in [30]). Let us now discuss *extensionality axioms*, i.e. axioms postulating the compatibility of extensional equality with application. To formulate them we first have to define extensional equality $=_\rho$ for an arbitrary type ρ, by

$$z_1 =_{\rho\to\sigma} z_2 := \forall x^\rho\, z_1x =_\sigma z_2x,$$

where $x =_{\mathrm{nat}} y$ is shorthand for $\mathrm{atom}(=x\,y)$ and $=\colon \mathrm{nat}\to\mathrm{nat}\to\mathrm{boole}$ is the characteristic function of equality on nat; $x =_{\mathrm{boole}} y$ is defined similarly. Then we extend the notion of a derivation term by constants for the extensionality axioms, as follows.

($\mathrm{Ext}_{\rho\sigma}$) For any types ρ,σ

$$\mathrm{Ext}_{\rho\sigma}\colon \forall x,y,z.x =_\rho y \to zx =_\sigma zy$$

is a derivation term with $\mathrm{FA}(\mathrm{Ext}_{\rho\sigma}) = \emptyset$.

(Ext_f) For any function symbol $f \in \mathcal{F}$

$$\mathrm{Ext}_f : \forall \vec{x} \forall \vec{y}. x_1 =_{\rho_1} y_1 \to \ldots \to x_m =_{\rho_m} y_m \to f(\vec{x}) =_\sigma f(\vec{y})$$

is a derivation term with $\mathrm{FA}(\mathrm{Ext}_f) = \emptyset$.

Clearly $\mathrm{FV}(\mathrm{Ext}_{\rho\sigma}) := FV(\mathrm{Ext}_f) := \emptyset$. The resulting system of arithmetic is called E–HA^ω in [30]. Below we discuss some relationships between the neutral and the extensional system.

1. E–HA^ω is conservative over N–HA^ω with respect to formulas containing variables (free or bound) of types of level ≤ 1 only. This can be seen by introducing logical relations (called *heriditary extensionality* in [30], pp. 155)

$$n \sim_{\mathrm{nat}} m := n = m,$$
$$f \sim_{\rho\to\sigma} g := \forall x, y. x \sim_\rho y \to fx \sim_\sigma gy.$$

In N–HA^ω the following facts can be proved easily.

(1) $\sim_\rho$ is symmetric and transitive and hence $x \sim_\rho y \to x \sim_\rho x$.

(2) $\vec{x} \sim \vec{y} \to r \sim r[\vec{y}/\vec{x}]$ for each term r with $\mathrm{FV}(r) \subseteq \vec{x}$.

For every formula A let $A^\sim$ be obtained from A by relativizing all quantifiers $\forall x^\rho$ and $\exists^* x^\rho$ to $x \sim_\rho x$.

(3) $x \sim_\rho x \to y \sim_\rho y \to (x \sim_\rho y \leftrightarrow (x =_\rho y)^\sim)$.

Using (2) and (3) we can show that to every derivation $d: A$ in E–HA^ω there exists a derivation $d^\sim : A^\sim$ in N–HA^ω with $\mathrm{FA}(d^\sim) = \{B^\sim | B \in \mathrm{FA}(d)\} \cup \{x \sim x | x \in \mathrm{FV}(d)\}$. If d is an extensionality axiom we use (3). If d is an $\forall$-elimination we use (2). All other cases are easy. The conservativity result now follows since in N–HA^ω we can prove $x \sim_\rho x$ for types ρ of level ≤ 1 and hence $A \leftrightarrow A^\sim$ if A contains quantifiers of type level ≤ 1 only. The proof also shows that for universal quantifiers in negative position and existential quantifiers in positive positions we can remove the restriction on types.

2. Howard has shown in [30], Appendix B that the formula

$$\forall x, y, u \exists z. xz = yz \to ux = uy$$

with $x, y : \mathrm{nat} \to \mathrm{nat}$ and $u : (\mathrm{nat} \to \mathrm{nat}) \to \mathrm{nat}$ (which is equivalent to $\mathrm{Ext}_{\mathrm{nat}\to\mathrm{nat},\mathrm{nat}}$) is an example of an $\forall\exists$-formula which is provable in E–HA^ω such that the corresponding $\forall\exists^*$-formula

$$\forall x, y, u \exists^* z. xz = yz \to ux = uy$$

is not provable in E–HA^ω (since the latter is equivalent to the Dialectica Interpretation of $\mathrm{Ext}_{\mathrm{nat}\to\mathrm{nat},\mathrm{nat}}$ this shows that the extensionality axioms are not Dialectica interpretable). Therefore E–HA^ω is not closed under Markov's rule, i.e. for quantifier free A we have that E–$HA^\omega \vdash \neg\neg\exists^* xA$ (note that $\vdash \neg\neg\exists^* xA \leftrightarrow \exists A$) does *not* imply E–$HA^\omega \vdash \exists^* xA$ in general. In the next chapter we will show that N–HA^ω *is* closed under Markov's rule.

Part II: Computational content of proofs

8 A–translation with program extraction and the direct method

As is well known a proof of a $\forall\exists$–theorem with a quantifier–free kernel — where $\exists$ is viewed as defined by $\neg\forall\neg$ — can be used as a program. We describe a "direct method" to use such a proof as a program, and compare it with Harvey Friedman's A–translation [10] followed by the well–known program extraction from constructive proofs.

The arguments presented work only for proofs not involving extensionality axioms.

8.1 Kreisel's counterexample

First note that a classical proof of $\forall x\exists y\, A$ generally does *not* yield a program to compute y from x. The reason for this is that there might be a universal quantifier $\forall z$ right after $\exists y$, i.e. after $\neg\forall y\neg$, and this makes it possible that an assumption

$$\forall y\neg\forall z\, B$$

is instantiated with a non–constant term containing critical variables which are bound later by $\forall z$.

It is well known that this is not just a technical difficulty: if T denotes Kleene's T–predicate, then

$$\forall n\exists m\forall k.T(n,n,k) \to T(n,n,m)$$

is trivially provable even in minimal logic (with $\exists m$ defined as $\neg\forall m\neg$, i.e. in classical logic), but there is no computable function f satisfying

$$\forall n,k.T(n,n,k) \to T(n,n,f(n)),$$

for then $\exists k\, T(n,n,k)$ would be decidable: it would be true if and only if $T(n,n,f(n))$ holds.

Hence in the rest of this section we will only consider formulas of the form $\forall x\exists y\, A$ with A quantifier–free.

8.2 The direct method

We first describe a "direct method" (cf. [23]) to extract the computational content from a classical proof.

By a Π–*formula* we mean a formula built without the strong existential quantifier $\exists^*$, which has no (universal) quantifier in premises of implications. For instance any Horn formula $\forall\vec{x}.P_1(\vec{x}) \to \cdots \to P_n(\vec{x}) \to Q(\vec{x})$ is a Π–formula, but

$$\forall x, y, u.(\forall z.xz = yz) \to ux = uy$$

is not. Clearly every Π–formula is equivalent (in minimal logic) to a conjunction of formulas $\forall C$ where C is quantifier–free and without $\wedge$. So from now on we will assume that Π–formulas are of this form.

A derivation d is called a *refutation of* Π–*assumptions* if d derives a closed false atom from assumptions $\mathrm{FA}(d) = \{v_1{:}\,\forall C_1, \ldots, v_n{:}\,\forall C_n\}$ where each C_i is quantifier–free.

Now let d be a refutation of Π–assumptions. We may assume $\mathrm{FV}(d) = \emptyset$ (if not, substitute arbitrary closed terms for the free variables in d). Next we can normalize d. Let $d{\downarrow}$ be the result. Again $d{\downarrow}$ is a refutation of Π–assumptions. We then can read off from $d{\downarrow}$ a list $|d|$ of closed terms called the "first instance" of $d{\downarrow}$ (cf. [23]) such that one of the Π–assumptions is false at $|d|$. To make this notion easier to understand let us restrict the general situation slightly. A closed quantifier–free formula B is *true* respectively *false* if t_B normalizes to the boolean constant true respectively false. A closed Π–formula $\forall\vec{x}\,C$ is true iff for all closed terms $\vec{t}$ the formula $C[\vec{t}/\vec{x}]$ is true.

Now let d be a refutation of Π–assumptions. We may assume $\mathrm{FV}(d) = \emptyset$ (if not, substitute arbitrary closed terms for the free variables in d). Next we can normalize d. Let $d{\downarrow}$ be the result. Again $d{\downarrow}$ is a refutation of Π–assumptions. We then can read off from $d{\downarrow}$ a list of closed terms called the "first instance" of $d{\downarrow}$ (cf. [23]). To make this notion easier to understand let us restrict the general situation slightly.

Let $d{:}\,\bot$ be a normal derivation with $\mathrm{FV}(d) = \emptyset$ of $\bot$ from assumptions

$$\begin{aligned} &u{:}\,\forall\vec{y}.B_1[\vec{y}] \to \ldots \to B_m[\vec{y}] \to \bot, \\ &v_1{:}\,\forall C_1, \ldots, v_n{:}\,\forall C_n \end{aligned}$$

where $\forall C_1, \ldots, \forall C_n$ are true closed Π–formulas. We define a list $|d|$ of closed terms, called the *first instance* of d, such that $B_1[|d|], \ldots, B_m[|d|]$ are true. $|d|$ is defined by induction on d. Since d is normal and $\mathrm{FV}(d) = \emptyset$ it does not contain axioms (exept the truth axiom, which is a closed Π–formulas and hence may be assumed to be among the Π–assumptions $\forall C_i$). To see this recall that the normal form of any closed term of type nat is of the form $S(S(S\ldots(S0)\ldots))$ and of any closed term of type boole is either true or false; hence all induction axioms unfold. Therefore d is of the form

$$w\vec{s}d_1 \ldots d_k,$$

where $\vec{s}$ are closed terms and $d_1, \ldots, d_k$ are derivations of closed quantifier–free formulas. We distinguish two cases.

1. $d_1, \ldots, d_k$ derive only true formulas (which can be decided, since the formulas are quantifier–free and closed). Then w cannot be one of the v_i since all $\forall C_i$ are true. Hence $d = u\vec{s}d_1 \ldots d_k$ and the d_i derive $B_i[\vec{s}]$. So let $|d| := \vec{s}$.

2. There is a minimal i such that d_i derives a false formula, $A_1 \to \cdots \to A_{n_i} \to \bot$ say. Then $A_1 \ldots, A_{n_i}$ are true. Without loss of generality we may assume that $d_i = \lambda w_1^{A_1} \ldots \lambda w_{n_i}^{A_{n_i}}\, e$ where $e\colon \bot$ contains assumptions among
$$\begin{aligned}
&u\colon \forall\vec{y}.B_1[\vec{y}] \to \ldots \to B_m[\vec{y}] \to \bot,\\
&v_1\colon \forall C_1, \ldots, v_n\colon \forall C_n,\\
&w_1\colon A_1, \ldots, w_{n_i}\colon A_{n_i}.
\end{aligned}$$
Therefore we can recursively define $|d| := |e|$.

Hence from a proof d of $\forall\vec{x}\exists\vec{y} \bigwedge_i B_i[\vec{x},\vec{y}]$ from true Π–assumptions with B_i quantifier–free we can obtain the following algorithm to compute for any $\vec{r}$ an $\vec{s}$ such that all $B_i[\vec{r},\vec{s}]$ hold. First instanciate d with $\vec{r}$, i.e. form $d\vec{r}\colon \exists\vec{y} \bigwedge_i B_i[\vec{r},\vec{y}]$. Since $\exists$ is $\neg\forall\neg$, we have $d\vec{r}u\colon \bot$ with $u\colon \forall\vec{y}.B_1[\vec{r},\vec{y}] \to \ldots \to B_m[\vec{r},\vec{y}] \to \bot$ a new Π–assumption (of a false formula!). Now normalize $d\vec{r}u$. From its normal form $(d\vec{r}u)\!\downarrow$, which is a refutation from Π–assumptions, we can read off the first instance $|(d\vec{r}u)\!\downarrow|$ of $(d\vec{r}u)\!\downarrow$. These are closed terms $\vec{s}$ such that all $B_i[\vec{r},\vec{s}]$ are true.

It might seem that instead of the method described, which chooses the branch to follow by checking whether some quantifier–free formulas are false or true, one could alternatively look for an occurrence of the false Π–assumption u in the proof whose arguments do not contain u any more. However, in our general case where Π–assumptions (and not just Horn formulas) are allowed these arguments may contain free assumption variables bound later (by $\to^+$) in the proof, and so we cannot conclude that all arguments of u derive true formulas. If, however, we restrict attention to the special case where we only allow Horn–formulas as assumptions, then this phenomenon cannot happen and we have a variant of the direct method.

Some comments are to be made here.

1. In principle, of course, we could replace any quantifier–free formula C by the atomic formula $\mathrm{atom}(t_C)$. However, this introduces quite a lot of somewhat artificial boolean functions, which makes the proof much harder to read and work with.

2. If we are prepared to apply some preparatory "pruning"–step to our derivation then we may assume that such additional assumptions v are always true. For if we had a $v\colon A$ with A false which is bound later in the proof yielding $A \to C$, then we could replace the whole subproof above this occurrence of $A \to C$ by a derivation of $A \to C$

using ex–falso–quodlibet. If we do this preparatory step first, then the variant of the direct method can be applied to the general case of Π–assumptions as well.

8.3 A–translation

We now describe Friedman's A–translation from [10]. Let A be an arbitrary but fixed formula. The A–translation B^A of a formula B is obtained by replacing any atomic subformula P of B by $(P \to A) \to A$.

Note that any derivation d of some formula B from assumptions $C_1, \dots, C_n$ becomes after the A–translation a derivation of B^A from $C_1^A, , \dots, C_n^A$. To see this recall that our logical rules are those of minimal logic and hence give no extra treatment to falsity. Also the axiom schemes (exept the truth axiom, which can be viewed as a Π–assumption) remain instances of the same axiom scheme after the A–translation. E.g. boolean induction

$$B[\mathrm{true}/p] \to B[\mathrm{false}/p] \to \forall p\, B$$

is translated into

$$B^A[\mathrm{true}/p] \to B^A[\mathrm{false}/p] \to \forall p\, B^A,$$

which again is an instance of boolean induction.

Let us look at what happens with Π–assumptions under the A–translation. As in §8.2 we may assume that all formulas considered do not contain $\wedge$.

Lemma 1. *For any quantifier–free formula C we can find a derivation $d: C \to C^A$.*

Proof by induction on C. Let $C \equiv B_1 \to \dots \to B_m \to R$ with an atom R. We must derive

$$(\vec{B} \to R) \to \vec{B}^A \to (R \to A) \to A.$$

So assume

$$\begin{aligned} &\tilde{u}: \vec{B} \to R, \\ &\tilde{v}_i: \vec{B}_i^A, \\ &w: R \to A. \end{aligned}$$

We must show A.

Case $u_i^-: \neg B_i$ for some i. Let $B_i \equiv \vec{C}_i \to P_i$ with atoms P_i. Then

$$\tilde{v}_i: \vec{C}_i^{\,A} \to (P_i \to A) \to A$$

and we have

$$\begin{aligned} &e_{ij}[u_i^-]: \equiv \mathrm{Stab}_{C_{ij}} \lambda v^{\neg C_{ij}}.u_i^- \lambda \vec{u}^{\vec{C}_i}.\mathrm{Efq}_{P_i}(v u_j): C_{ij}, \\ &e_i[u_i^-]: \equiv \lambda w_i^{P_i}.u_i^- \lambda \vec{u}^{\vec{C}_i} w_i: \neg P_i. \end{aligned}$$

By IH we have $d_{C_{ij}}\colon C_{ij} \to C_{ij}^A$. Hence

$$\tilde{v}_i(d_{C_{i1}} e_{i1}) \dots (d_{C_{in_i}} e_{in_i})(\lambda w_i^{P_i}.\mathrm{Efq}_A(e_i w_i))\colon A.$$

Case $u_i^+\colon B_i$ for all i. Then

$$w(\tilde{u} u_1^+ \dots u_m^+)\colon A$$

The extracted terms for this derivation are

$$\begin{aligned} d^{\mathrm{ets}} \equiv \lambda \vec{x}_1, \dots, \vec{x}_m, \vec{z}.\ & \textbf{if} \quad \neg B_1 \quad \textbf{then} \quad \vec{x}_1 d^{\mathrm{ets}}_{C_{11}} \dots d^{\mathrm{ets}}_{C_{1n_1}} \vec{0} \quad \textbf{else} \\ & \dots \\ & \textbf{if} \quad \neg B_m \quad \textbf{then} \quad \vec{x}_m d^{\mathrm{ets}}_{C_{m1}} \dots d^{\mathrm{ets}}_{C_{mn_m}} \vec{0} \quad \textbf{else} \\ & \vec{z} \quad \textbf{fi} \dots \textbf{fi}, \end{aligned}$$

where $\vec{x}_i, \vec{z}$ are the lists of variables associated with $\tilde{v}_i\colon B_i^A, w\colon R \to A$. □

Here we have used case splitting according to the quantifier free formulas B_i which is admissible by the remark at the end of §7.

If we want to use the A–translation to extract the computational content from a classical proof we have to choose a particular A involving the strong existential quantifier.

Lemma 2. *Let* $B_i[\vec{x}, \vec{y}]$ *be quantifier–free formulas and* $A[\vec{x}] := \exists^* \vec{y} \bigwedge_i B_i[\vec{x}, \vec{y}]$. *Then we can find a derivation of* $(\forall \vec{y}. B_1[\vec{x}, \vec{y}] \to \dots \to B_m[\vec{x}, \vec{y}] \to \bot)^{A[\vec{x}]}$.

Proof. Let $\vec{y}$ be given and assume $\tilde{v}_i\colon \vec{B}_i^A$ and $w\colon \bot \to A$. We must show A.

Case $u_i^-\colon \neg B_i$ for some i. Let $B_i \equiv \vec{C}_i \to P_i$ with atoms P_i. Then

$$\tilde{v}_i\colon \vec{C}_i^{\,A} \to (P_i \to A) \to A$$

and we have

$$\begin{aligned} e_{ij}[u_i^-]\colon &\equiv \mathrm{Stab}_{C_{ij}} \lambda v^{\neg C_{ij}}.u_i^- \lambda \vec{u}^{\vec{C}_i}.\mathrm{Efq}_{P_i}(v u_j)\colon C_{ij}, \\ e_i[u_i^-]\colon &\equiv \lambda w_i^{P_i}.u_i^- \lambda \vec{u}^{\vec{C}_i} w_i\colon \neg P_i. \end{aligned}$$

Using $d_{C_{ij}}\colon C_{ij} \to C_{ij}^A$ from Lemma 1 we obtain

$$\tilde{v}_i(d_{C_{i1}} e_{i1}) \dots (d_{C_{in_i}} e_{in_i})(\lambda w_i^{P_i}.\mathrm{Efq}_A(e_i w_i))\colon A.$$

Case $u_i^+\colon B_i$ for all i. Then

$$\exists^+ \vec{y} \langle u_1^+, \dots, u_m^+ \rangle\colon A$$

The extracted terms for this derivation are

$$\begin{aligned} d^{\mathrm{ets}} \equiv \lambda \vec{y}, \vec{x}_1, \dots, \vec{x}_m, \vec{z}.\ & \textbf{if} \quad \neg B_1 \quad \textbf{then} \quad \vec{x}_1 d^{\mathrm{ets}}_{C_{11}} \dots d^{\mathrm{ets}}_{C_{1n_1}} \vec{0} \quad \textbf{else} \\ & \dots \\ & \textbf{if} \quad \neg B_m \quad \textbf{then} \quad \vec{x}_m d^{\mathrm{ets}}_{C_{m1}} \dots d^{\mathrm{ets}}_{C_{mn_m}} \vec{0} \quad \textbf{else} \\ & \vec{y} \quad \textbf{fi} \dots \textbf{fi}, \end{aligned}$$

where $\vec{x}_i, \vec{z}$ are the lists of variables associated with $\tilde{v}_i\colon B_i^A, w\colon \bot \to A$. □

Theorem (Friedman). *For any derivation*

$$d[u\colon \forall\vec{y}.B_1[\vec{x},\vec{y}] \to \ldots \to B_m[\vec{x},\vec{y}] \to \bot, v_1\colon \forall C_1, \ldots, v_n\colon \forall C_n]\colon \bot$$

with B_i, C_j quantifier–free we can find a derivation

$$d^{\mathrm{tr}}[v_1\colon \forall C_1, \ldots, v_n\colon \forall C_n]\colon \exists^*\vec{y} \bigwedge_i B_i[\vec{x},\vec{y}].$$

Proof. Let $A[\vec{x}] := \exists^*\vec{y} \bigwedge_i B_i[\vec{x},\vec{y}]$ and consider the $A[\vec{x}]$–translation

$$\begin{aligned} & d^{A[\vec{x}]}[u'\colon (\forall\vec{y}.B_1[\vec{x},\vec{y}] \to \ldots \to B_m[\vec{x},\vec{y}] \to \bot)^{A[\vec{x}]}, v_1'\colon (\forall C_1)^{A[\vec{x}]}, \ldots v_n'\colon (\forall C_n)^{A[\vec{x}]}] \\ & \quad\colon (\bot \to A[\vec{x}]) \to A[\vec{x}] \end{aligned}$$

of d, obtained by just changing some formulas. By Lemma 1 we have

$$d_{v_i}[v_i\colon \forall C_i]\colon (\forall C_i)^{A[\vec{x}]}.$$

By Lemma 2 (now using the particular choice of $A[\vec{x}]$) the $A[\vec{x}]$–translation of the assumption u is provable without assumptions:

$$d_u\colon (\forall\vec{y}.B_1[\vec{x},\vec{y}] \to \ldots \to B_m[\vec{x},\vec{y}] \to \bot)^{A[\vec{x}]}$$

Substituting $d_{v_i}[v_i\colon \forall C_i]$ for v_i' and d_u for u' we obtain

$$d^{\mathrm{tr}}[v_1\colon \forall C_1, \ldots, v_n\colon \forall C_n]\colon \exists^*\vec{y} \bigwedge_i B_i[\vec{x},\vec{y}],$$

where

$$d^{\mathrm{tr}} \equiv d^{A[\vec{x}]}[d_u, d_{v_1}, \ldots, d_{v_n}]\mathrm{Efq}_{A[\vec{x}]}. \qquad \square$$

Having obtained a proof d^{tr} of an existential formula $\exists^*\vec{y} \bigwedge_i B_i[\vec{x},\vec{y}]$ we can then apply the general method of extracting terms to this proof. It yields

$$(d^{\mathrm{tr}})^{\mathrm{ets}} \equiv (d^{A[\vec{x}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}, d_{v_1}^{\mathrm{ets}}, \ldots, d_{v_n}^{\mathrm{ets}}]\vec{0}, \tag{1}$$

since extracting terms commutes with substitution.

Note that there are many ways to use Friedman's method to extract a term from a given arithmetical proof of a weak existential formula.

1. The most straightforward way is to prove the formula without any assumptions. This means that we are not allowed to use lemmata, and hence that such a proof tends to be rather long, and difficult to produce.

2. TThe next straightforward way is to pack all Π–Lemmata used in the proof (and proved explicitely in 1 above) into purely generalized atoms $\forall\vec{x}\,\mathrm{atom}(t)$. However, this means that we have to introduce rather complex boolean terms, which will later show up in the proof as tests for case distinctions.

8.4 Comparison

We now prove that the value of the extracted terms $(d^{\mathrm{tr}})^{\mathrm{ets}}$ when instanciated with a list $\vec{r}$ of closed terms is in fact the same as the result of the direct method described in §8.2.

So consider again the situation of Friedman's Theorem, i.e. a derivation

$$d[u\colon \forall\vec{y}.B_1[\vec{x},\vec{y}] \to \ldots \to B_m[\vec{x},\vec{y}] \to \bot, v_1\colon \forall C_1, \ldots, v_n\colon \forall C_n]\colon \bot$$

with B_i, C_j quantifier–free. We just observed that the program $(d^{\mathrm{tr}})^{\mathrm{ets}}$ extracted from the translated derivation has the form (1) above. Let us try to understand how this program works. First, $(d^{A[\vec{x}]})^{\mathrm{ets}}$ closely follows the structure of d. The reason is that $d^{A[\vec{x}]}$ differs from d only with respect to the formulas affixed, and when forming the extracted terms this affects only the types and the arities of the lists of object variables associated with assumption variables.

In order to comprehend $d_{v_i}^{\mathrm{ets}}$ and d_u^{ets} let us have a second look at the proofs of Lemma 1 and 2. First note that $d_{v_i}[v_i\colon \forall C_i]\colon (\forall C_i)^{A[\vec{x}]}$ is obtained from $d_i\colon C_i \to C_i^{A[\vec{x}]}$ constructed in the proof of Lemma 1 by

$$d_{v_i} \equiv \lambda\vec{y}_i.d_i(v_i\vec{y}_i).$$

Since v_i has type $\forall C_i$, which is a Harrop formula, we have $d_{v_i}^{\mathrm{ets}} \equiv \lambda\vec{y}_i\, d_i^{\mathrm{ets}}$. Now from the proof of Lemma 1 we obtain

$$\begin{aligned} d_i^{\mathrm{ets}} \equiv \lambda\vec{x}_1,\ldots,\vec{x}_m,\vec{z}.\ &\textbf{if}\quad \neg B_1 \quad \textbf{then}\quad \vec{x}_1 d_{C_{11}}^{\mathrm{ets}} \ldots d_{C_{1n_1}}^{\mathrm{ets}}\vec{0}\quad \textbf{else}\\ &\ldots\\ &\textbf{if}\quad \neg B_m \quad \textbf{then}\quad \vec{x}_m d_{C_{m1}}^{\mathrm{ets}} \ldots d_{C_{mn_m}}^{\mathrm{ets}}\vec{0}\quad \textbf{else}\\ &\vec{z}\quad \textbf{fi}\ldots\textbf{fi}, \end{aligned} \tag{2}$$

where $C_i \equiv B_1 \to \ldots \to B_m \to R$ with $B_i \equiv \vec{C}_i \to P_i$ and $\vec{x}_i, \vec{z}$ are the lists of variables associated with $\tilde{v}_i, w$. Furthermore, $d_{C_{ij}}^{\mathrm{ets}}$ are the extracted terms of derivations $d_{C_{ij}}\colon C_{ij} \to C_{ij}^{A[\vec{x}]}$ constructed by previous applications of Lemma 1. Similarly

$$\begin{aligned} d_u^{\mathrm{ets}} \equiv \lambda\vec{y},\vec{x}_1,\ldots,\vec{x}_m,\vec{z}.\ &\textbf{if}\quad \neg B_1 \quad \textbf{then}\quad \vec{x}_1 d_{C_{11}}^{\mathrm{ets}} \ldots d_{C_{1n_1}}^{\mathrm{ets}}\vec{0}\quad \textbf{else}\\ &\ldots\\ &\textbf{if}\quad \neg B_m \quad \textbf{then}\quad \vec{x}_m d_{C_{m1}}^{\mathrm{ets}} \ldots d_{C_{mn_m}}^{\mathrm{ets}}\vec{0}\quad \textbf{else}\\ &\vec{y}\quad \textbf{fi}\ldots\textbf{fi}, \end{aligned} \tag{3}$$

where $B_i \equiv \vec{C}_i \to P_i$ and $\vec{x}_i, \vec{z}$ are the lists of variables associated with $\tilde{v}_i, w$. Furthermore, $d_{C_{ij}}^{\mathrm{ets}}$ are the extracted terms of derivations $d_{C_{ij}}\colon C_{ij} \to C_{ij}^{A[\vec{x}]}$ constructed by previous applications of Lemma 1.

This analysis makes it possible to prove that the value of the extracted terms when instanciated with a list $\vec{r}$ of closed terms is in fact the same as the result of the direct method described in §8.2 to read off the first instance provided by the instanciated derivation

$$\bar{d}[\bar{u}\colon \forall\vec{y}.B_1[\vec{r},\vec{y}] \to \ldots \to B_m[\vec{r},\vec{y}] \to \bot, v_1\colon \forall C_1, \ldots, v_n\colon \forall C_n]\colon \bot$$

Below we will show the following

Claim. *For any normal derivation*

$$e[\bar{u}\colon \forall\vec{y}.B_1[\vec{r},\vec{y}] \to \ldots \to B_m[\vec{r},\vec{y}] \to \bot, v_1\colon \forall C_1,\ldots,v_n\colon \forall C_n]\colon \bot$$

with $\mathrm{FV}(e)=\emptyset$ *we have*

$$|e| = [\![(e^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}/\vec{x}], d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\vec{0}]\!].$$

We then obtain that the instanciation of the extracted terms (1) with $\vec{r}$ for $\vec{x}$, i.e.

$$(d^{A[\vec{x}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}, d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\vec{0}[\vec{r}/\vec{x}] \equiv (d[\vec{r}/\vec{x}]^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}/\vec{x}], d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\vec{0}$$

has as its value the list of closed terms which is the first instance of the instanciated derivation $d[\vec{r}/\vec{x}]$, i.e. $|d[\vec{r}/\vec{x}]{\downarrow}|$. For by the claim we have

$$\begin{aligned}|d[\vec{r}/\vec{x}]{\downarrow}| &= [\![((d[\vec{r}/\vec{x}]{\downarrow})^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}/\vec{x}], d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\vec{0}]\!]\\ &= [\![((d[\vec{r}/\vec{x}]^{A[\vec{r}]})^{\mathrm{ets}}){\downarrow}[d_u^{\mathrm{ets}}[\vec{r}/\vec{x}], d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\vec{0}]\!]\\ &= [\![(d[\vec{r}/\vec{x}]^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}/\vec{x}], d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\vec{0}]\!],\end{aligned}$$

since normalization (i.e. $\beta\eta\exists^* R$–conversion) commutes with $A[\vec{r}]$–translation and the formation of extracted terms.

It remains to prove the claim. We use induction on e. Since e is normal, it must be of the form $e = w\vec{s}e_1\ldots e_k$ with $w\in\{\bar{u},v_1,\ldots,v_n\}$.

Case 1. $e_1,\ldots,e_k$ derive only true formulas. Then $w=\bar{u}$, $k=m$ and the e_i derive $B_i[\vec{r},\vec{s}]$. By definition $|e| := \vec{s}$. Furthermore

$$\begin{aligned}&(e^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}/\vec{x}], d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\vec{0}\\ &\equiv d_u^{\mathrm{ets}}[\vec{r}/\vec{x}]\vec{s}(e_1^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}/\vec{x}], d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\ldots(e_m^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}/\vec{x}], d_{v_1}^{\mathrm{ets}},\ldots,d_{v_n}^{\mathrm{ets}}]\vec{0}\\ &=_{\beta R} \vec{s}\end{aligned}$$

by the form (3) of d_u^{ets}, since all $B_i[\vec{r},\vec{s}] \equiv (\vec{B}_i \to P_i)[\vec{r},\vec{s}]$ are true.

Case 2. There is a minimal i such that e_i derives a false formula, $D_{i1}[\vec{s}] \to \cdots \to D_{in_i}[\vec{s}] \to \bot$ say. Then $D_{i1}[\vec{s}],\ldots,D_{in_i}[\vec{s}]$ are true. Without loss of generality we may assume that $e_i = \lambda w_1^{D_{i1}[\vec{s}]}\ldots\lambda w_m^{D_{in_i}[\vec{s}]} f$ where $f\colon\bot$ contains assumptions among

$$\begin{aligned}&\bar{u}\colon \forall\vec{y}.B_1[\vec{r},\vec{y}] \to \ldots \to B_m[\vec{r},\vec{y}] \to \bot,\\ &v_1\colon \forall C_1,\ldots,v_n\colon \forall C_n,\\ &w_1\colon D_{i1},\ldots,w_m\colon D_{in_i}.\end{aligned}$$

Therefore by definition $|e| := |f|$. Furthermore using the notation $d_{w_j}[w_j]\colon D_{ij}^{A[\vec{r}]}$ for the derivation obtained by applying Lemma 1 to D_{ij} we have

$$
\begin{aligned}
&(e^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}], d_{v_1}^{\mathrm{ets}}, \dots, d_{v_n}^{\mathrm{ets}}]\vec{0} \\
&\equiv \begin{cases} d_u^{\mathrm{ets}}[\vec{r}]\vec{s}\,(e_1^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}], d_{v_1}^{\mathrm{ets}}, \dots, d_{v_n}^{\mathrm{ets}}] \dots (e_k^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}], d_{v_1}^{\mathrm{ets}}, \dots, d_{v_n}^{\mathrm{ets}}]\vec{0} & \text{if } w = \bar{u} \\ d_{v_i}^{\mathrm{ets}}\vec{s}\,(e_1^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}], d_{v_1}^{\mathrm{ets}}, \dots, d_{v_n}^{\mathrm{ets}}] \dots (e_k^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}], d_{v_1}^{\mathrm{ets}}, \dots, d_{v_n}^{\mathrm{ets}}]\vec{0} & \text{if } w = v_i \end{cases} \\
&=_{\beta R} (e_i^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}], d_{v_1}^{\mathrm{ets}}, \dots, d_{v_n}^{\mathrm{ets}}] d_{w_1}^{\mathrm{ets}} \dots d_{w_{n_i}}^{\mathrm{ets}}\vec{0} \quad \text{by (3) and (2), respectively} \\
&=_{\beta} (f^{A[\vec{r}]})^{\mathrm{ets}}[d_u^{\mathrm{ets}}[\vec{r}], d_{v_1}^{\mathrm{ets}}, \dots, d_{v_n}^{\mathrm{ets}}, d_{w_1}^{\mathrm{ets}}, \dots, d_{w_{n_i}}^{\mathrm{ets}}]\vec{0},
\end{aligned}
$$

so the claim follows from the IH. □

9 The root example; refinements

In applications it will be important to produce extracted terms with as few as possible case distinctions, and also that the case distinctions should be over as simple as possible boolean terms. The following example will show that such improvements are indeed necessary.

Let $f\colon \mathrm{nat} \to \mathrm{nat}$ be an unbounded function with $f(0) = 0$. Then we can prove

$$\forall n \exists m. f(m) \leq n < f(m+1).$$

If e.g. $f(m) = m^2$, then this formula expresses the existence of an integer square root $m := [\sqrt{n}]$ for any n. More formally we can prove

$$\forall n \exists m. \neg n < f(m) \wedge n < f(m+1) \tag{1}$$

from the assumptions

$$v_1\colon \forall n\, \neg n < f(0), \qquad v_2\colon \forall n\, n < f(g(n)).$$

Here $<^{\mathrm{nat} \to \mathrm{nat} \to \mathrm{boole}}$ is the characteristic function of the natural ordering of the natural numbers and $r < s$ denotes $\mathrm{atom}(<rs)$. We expressed $f(m) \leq n$ by $\neg n < f(m)$ and $f(0) = 0$ by $\forall n\, \neg n < f(0)$ to keep the formal proof as simple as possible. In order to have Π–assumptions we had to express the unboundedness of f by a witnessing function g. First note that the Π–assumptions v_1 and v_2 are not closed as required previously. But this is no problem since A–translation and program extraction clearly also work in this case: we just get a program containing f and g as parameters.

Now let us prove (1). Let n be given and assume

$$u\colon \forall m. \neg n < f(m) \to n < f(m+1) \to \bot.$$

We have to show $\bot$. From v_1 and u we inductively get $\forall m\, \neg n < f(m)$. For $m := g(n)$ this yields a contradiction to v_2.

The derivation term corresponding to this proof is

$$d := \mathrm{Ind}_{m, \neg n < f(m)} n (v_1 n) u (g(n)) (v_2 n)\colon \bot.$$

Now let

$$A := \exists^* m. \neg n < f(m) \wedge n < f(m+1).$$

The program extracted from d is

$$(d^{\mathrm{tr}})^{\mathrm{ets}} \equiv (d^A)^{\mathrm{ets}}[d_u^{\mathrm{ets}}, d_{v_1}^{\mathrm{ets}}, d_{v_2}^{\mathrm{ets}}] 0\colon \mathrm{nat}.$$

$(d^A)^{\text{ets}}$ is the same as d exept that $\text{Ind}_{m,\neg n<f(m)}$ has to be replaced by

$$\lambda n\, R_{\text{nat},(\text{nat}\to\text{nat})\to\text{nat}\to\text{nat}}$$

(since $\tau((\neg n < f(m))^A) = \tau(((n < f(m) \to A) \to A) \to (\bot \to A) \to A) = (\text{nat} \to \text{nat}) \to \text{nat} \to \text{nat})$, and the assumption variables u, v_1, v_2 in d have to be replaced by (unary lists of) object variables

$$\begin{aligned}
&x_u\colon \text{nat} \to [(\text{nat} \to \text{nat}) \to \text{nat} \to \text{nat}] \to (\text{nat} \to \text{nat}) \to \text{nat} \to \text{nat},\\
&x_{v_1}\colon \text{nat} \to (\text{nat} \to \text{nat}) \to \text{nat} \to \text{nat},\\
&x_{v_2}\colon \text{nat} \to \text{nat} \to \text{nat}.
\end{aligned}$$

The subprograms $d_u^{\text{ets}}, d_{v_1}^{\text{ets}}, d_{v_2}^{\text{ets}}$ (of the same types as x_u, x_{v_1}, x_{v_2}) are given by

$$\begin{aligned}
d_u^{\text{ets}} \equiv \lambda m, x_1, x_2, k.\ &\textbf{if}\quad n < f(m) \quad \textbf{then}\quad x_1\text{id}0 \quad \textbf{else}\\
&\textbf{if}\quad \neg n < f(m+1) \quad \textbf{then}\quad x_2 0 \quad \textbf{else}\\
&m \quad \textbf{fifi},\\
d_{v_1}^{\text{ets}} \equiv \lambda n, x_1, k.\ &\textbf{if}\quad \neg n < f(0) \quad \textbf{then}\quad x_1 0 \quad \textbf{else}\\
&k \quad \textbf{fi},\\
d_{v_2}^{\text{ets}} \equiv \lambda n, k\, k.&
\end{aligned}$$

Hence the normal form of $(d^{\text{tr}})^{\text{ets}}$ is

$$\begin{aligned}
&R(\lambda x_1, k.\ \textbf{if}\quad \neg n < f(0) \quad \textbf{then}\quad x_1 0 \quad \textbf{else}\quad k \quad \textbf{fi})\\
&\quad(\lambda m, w_1, w_2, k.\ \textbf{if}\quad n < f(m) \quad \textbf{then}\quad w_1\text{id}0 \quad \textbf{else}\\
&\qquad\qquad\qquad\quad\ \textbf{if}\quad \neg n < f(m+1) \quad \textbf{then}\quad w_2 0 \quad \textbf{else}\\
&\qquad\qquad\qquad\quad\ m \quad \textbf{fifi})\\
&\quad(g(n))\\
&\quad(\lambda k\, k)\\
&\quad 0
\end{aligned}$$

Informally, $(d^{\text{tr}})^{\text{ets}} = H(g(n), \lambda k\, k, 0)$ where $H\colon \text{nat} \to (\text{nat} \to \text{nat}) \to \text{nat} \to \text{nat}$ is such that

$$\begin{aligned}
H(0, x_1, k) &= \textbf{if}\quad \neg n < f(0) \quad \textbf{then}\quad x_1 0 \quad \textbf{else}\quad k \quad \textbf{fi}\\
H(m+1, x_1, k) &= \textbf{if}\quad n < f(m) \quad \textbf{then}\quad H(m, \text{id}, 0) \quad \textbf{else}\\
&\ \ \textbf{if}\quad \neg n < f(m+1) \quad \textbf{then}\quad x_1 0 \quad \textbf{else}\\
&\ \ m \quad \textbf{fifi}
\end{aligned}$$

This program is correct, but it is unnecessarily complicated. We will now describe a refined A–translation which will simplify the type of the auxiliary function H as well as its if–then–else structure. The type reduction will be achieved by *not* replacing *all* atoms P

by $(P \to A) \to A$. This of course requires that atom is not the only relation symbol. Furthermore, to reduce case splitting we will construct "better" proofs of $C \to C^A$ for quantifier–free C without $\wedge$.

We first extend our arithmetical system by introducing relation symbols different from atom, and also reintroducing a special symbol $\bot$ for falsity. The reason for the latter addition is that otherwise e.g. the boolean induction axiom

$$\mathrm{atom}(\mathrm{true}) \to \mathrm{atom}(\mathrm{false}) \to \forall p\, \mathrm{atom}(p)$$

would translate into

$$\mathrm{atom}(\mathrm{true}) \to A \to \forall p\, \mathrm{atom}(p),$$

which certainly is not derivable. Formulas are now built from atomic formulas $\bot$, $\mathrm{atom}(t)$ and possibly other atoms $P(\vec{t})$ by means of $\to$, $\wedge$, $\forall$ and $\exists^*$.

We also extend the notion of a derivation term by adding the following clauses. $\neg\mathrm{F}\colon \neg\mathrm{atom}(\mathrm{false})$ is a derivation term with $\mathrm{FA}(\neg\mathrm{F}) := \mathrm{FV}(\neg\mathrm{F}) := \emptyset$, and $\mathrm{Efq}_{\mathrm{false}}\colon \bot \to \mathrm{atom}(\mathrm{false})$ is a derivation term with $\mathrm{FA}(\mathrm{Efq}_{\mathrm{false}}) := \mathrm{FA}(\mathrm{Efq}_{\mathrm{false}}) := \emptyset$. Note that by $\mathrm{Efq}_{\mathrm{false}}$ and the falsity axiom $\neg\mathrm{F}\colon \neg\mathrm{atom}(\mathrm{false})$ we have

$$\vdash \mathrm{atom}(\mathrm{false}) \leftrightarrow \bot.$$

$\mathrm{Stab}_{\mathrm{atom}}\colon \forall p. \neg\neg\mathrm{atom}(p) \to \mathrm{atom}(p)$ can again be proved easily by boolean induction, using the truth axiom in the case true, and the falsity axiom $\neg\mathrm{F}$ and $\mathrm{Efq}_{\mathrm{false}}$ in the case false. As before we can conclude $\vdash \neg\neg A \to A$ for formulas A built with $\bot$, $\to$, $\wedge$ and $\forall$ and containing atom as the only the relation symbol. $\bot \to A$ is derivable for any formula A containing atom as the only the relation symbol.

Let us call a relation symbol P *decidable* if there is a term t_P such that $\vdash \forall \vec{x}. P(\vec{x}) \leftrightarrow \mathrm{atom}(t_P \vec{x})$. Again a formula A is called decidable if there is a term t_A such that $\vdash A \leftrightarrow \mathrm{atom}(t_A)$. As before every quantifier–free formula containing only decidable relation symbols is decidable, and we can do case splitting according to decidable formulas A.

Now let L be a set of formulas. In our applications L will consist of the quantifier–free kernels of the lemmata $\forall C_i$, and in addition of the formula $B_1[\vec{x}, \vec{y}] \to \ldots \to B_m[\vec{x}, \vec{y}] \to \bot$, if our goal formula is $\forall \vec{x} \exists \vec{y} \bigwedge\!\!\!\bigwedge_i B_i[\vec{x}, \vec{y}]$.

The set of L*–critical* relation symbols is the smallest set satisfying the following condition.

> If $(\vec{C}_1 \to P_1) \to \ldots \to (\vec{C}_m \to P_m) \to R(\vec{t})$ is a positive subformula of an L–formula, and if for some i $P_i \equiv \bot$ or $P_i \equiv Q(\vec{s})$ for some L–critical relation symbol Q, then R is L–critical.

We now define an A–translation relative to L:

$$R(\vec{t})^A := \begin{cases} (R(\vec{t}) \to A) \to A, & \text{if } R \text{ is } L\text{–critical} \\ R(\vec{t}), & \text{otherwise} \end{cases}$$

$$\bot^A := A$$

$$(B \to C)^A := B^A \to C^A.$$

We will write $P \in \mathcal{C}_L$ if the atom P is of the form $R(\vec{t})$ for some L–critical R or $P \equiv \bot$.

A quantifier–free formula $B_1 \to \ldots \to B_m \to R$ will be called *L–relevant* if $R \in \mathcal{C}_L$.

Lemma 1*. *For any $B \in \mathrm{Neg}(L)$ and any $C \in \mathrm{Pos}(L)$ we can find the following derivations.*

(i) $d_C\colon C \to C^A$. Let $C \equiv \vec{B} \to R$. Then in case $R \in \mathcal{C}_L$ we need g_{B_i} for B_i L–relevant and f_{B_j} for B_j L–irrelevant, and in case $R \notin \mathcal{C}_L$ we need $f_{\vec{B}}$.

(ii) $e_{C,D}\colon ((C \to D) \to A) \to C^A$ for C L–relevant. Let $C \equiv \vec{B} \to R$. Then we need Cases_R if $R \not\equiv \bot$, and also g_{B_i} or f_{B_i} depending on whether B_i is L–relevant or not.

(iii) $f_B\colon B^A \to B$ for B L–irrelevant. Let $B \equiv \vec{C} \to R$. Then we need $d_{\vec{C}}$.

(iv) $g_B\colon B^A \to (B \to A) \to A$ for B L–relevant. Let $B \equiv \vec{C} \to R$. Then we need some $e_{C_i,D}$ for C_i L–relevant, and Cases_{C_j} and also d_{C_j} for C_j L–irrelevant.

Proof. Simultaneously by induction on the subformulas of L–formulas.

(i) Let $C \equiv \vec{B} \to R$. *Case* $R \in \mathcal{C}_L$, $R \not\equiv \bot$. We must derive

$$(\vec{B} \to R) \to \vec{B}^A \to (R \to A) \to A.$$

So assume

$$\begin{aligned} &u\colon \vec{B} \to R,\\ &v_i\colon B_i^A,\\ &w\colon R \to A. \end{aligned}$$

We must show A.

Let B_i for $i \in \{1, \ldots, k\}$ be relevant and B_j for $j \in \{k+1, \ldots, m\}$ be irrelevant. Assume $u_i\colon B_i$. Note that by IH(iii) $f_{B_j}\colon B_j^A \to B_j$ and hence $f_{B_j} v_j\colon B_j$. From $u\colon B_1 \to \ldots \to B_k \to B_{k+1} \to \ldots \to B_m \to R$ and $w\colon R \to A$ we get A. Now cancel $u_k\colon B_k$, yielding $B_k \to A$. Using the IH $g_{B_k}\colon B_k^A \to (B_k \to A) \to A$ and the assumption B_k^A we get A. Repeating this procedure we finally cancel $u_1\colon B_1$, yielding $B_1 \to A$. Using the IH $g_{B_1}\colon B_1^A \to (B_1 \to A) \to A$ and the assumption B_1^A we get A, as required. The derivation term is

$$d_C \equiv \lambda u, v_1, \ldots, v_m, w. g_{B_1} v_1 \lambda u_1^{B_1}. \cdots g_{B_k} v_k \lambda u_k^{B_k}. w \left(u u_1 \ldots u_k (f_{B_{k+1}} v_{k+1}) \ldots (f_{B_m} v_m) \right)$$

and we get

$$d_C^{\mathrm{ets}} \equiv \lambda \vec{x}_1, \ldots, \vec{x}_k, \vec{z}. g_{B_1}^{\mathrm{ets}} \vec{x}_1 (g_{B_2}^{\mathrm{ets}} \vec{x}_2 \ldots (g_{B_k}^{\mathrm{ets}} \vec{x}_k \vec{z}) \ldots),$$

where $\vec{x}_i, \vec{z}$ are the lists of variables associated with v_i, w.

$$\dfrac{\dfrac{(R\to A)\to(\neg R\to A)\to A \quad v\colon R\to A}{(\neg R\to A)\to A} \qquad \dfrac{\dfrac{u\colon (R\to D)\to A \qquad \dfrac{\dfrac{\perp\to D \qquad \dfrac{u_1\colon\neg R \quad u_2\colon R}{\perp}}{D}}{R\to D}}{A}}{\neg R\to A}}{A}$$

Figure 1

Case $R \equiv \perp$. Similarly, using Efq_A instead of $w\colon R \to A$. Then in d_C^{ets} we leave out $\lambda\vec{z}$ and replace $\vec{z}$ in the kernel by $\vec{0}$.

Case $R \notin \mathcal{C}_L$. Then all B_i are irrelevant. We must derive

$$(\vec{B} \to R) \to \vec{B}^A \to R.$$

So assume $u\colon \vec{B} \to R$ and $\vec{B}^A$. Using the IH $f_{B_i}\colon B_i^A \to B_i$ we obtain $\vec{B}$ and hence R. In this case $d_C^{\mathrm{ets}} \equiv \varepsilon$.

(ii) *Case* R with $R \in \mathcal{C}_L$, $R \not\equiv \perp$. Then $R^A \equiv (R \to A) \to A$, and we must derive

$$((R \to D) \to A) \to (R \to A) \to A.$$

So assume $u\colon (R \to D) \to A$ and $v\colon R \to A$. Then clearly A can be derived, using Cases_R and Efq_D; a derivation is given in Figure 1.

The derivation term is

$$e_{R,D} \equiv \lambda u, v.\mathrm{Cases}_R v \lambda u_1^{\neg R}.u\lambda u_2^R.\mathrm{Efq}_D(uu_1)$$

and we get

$$e_{R,D}^{\mathrm{ets}} \equiv \lambda\vec{x}, \vec{y}.\ \textbf{if} \quad R \quad \textbf{then} \quad \vec{y} \quad \textbf{else} \quad \vec{x} \quad \textbf{fi}.$$

Case R with $R \equiv \perp$. Then $R^A \equiv A$, and we must derive

$$((\perp \to D) \to A) \to A.$$

But this clearly can be done using Efq_D, and we have $e_{\perp,D}^{\mathrm{ets}} \equiv \lambda\vec{x}\,\vec{x}$.

$$\dfrac{\dfrac{\dfrac{B^A \to (B\to A)\to A \quad v\colon B^A}{(B\to A)\to A} \qquad \dfrac{\dfrac{u\colon((B\to C)\to D)\to A \qquad \dfrac{\dfrac{u_1\colon C\to D \qquad \dfrac{u_3\colon B\to C \quad u_2\colon B}{C}}{D}}{(B\to C)\to D}}{A}}{B\to A}}{A}}{(C\to D)\to A}$$

Figure 2

Case $B \to C$. We must derive

$$[((B \to C) \to D) \to A] \to B^A \to C^A$$

So assume $u\colon ((B \to C) \to D) \to A$ and $v\colon B^A$. We must show C^A. First note that we can derive $(C \to D) \to A$ from our assumptions, using the fact that by IH we can derive $B^A \to (B \to A) \to A$ (with g_B or f_B, depending on whether B is relevant or not). A derivation is given in Figure 2. But then the claim follows, since by IH

$$e_{C,D}\colon ((C \to D) \to A) \to C^A.$$

The derivation term is

$$\lambda u, v.e_{C,D}\lambda u_1.g_B v\lambda u_2.u\lambda u_3.u_1(u_3u_2) \quad \text{or} \quad \lambda u, v.e_{C,D}\lambda u_1.u\lambda u_3.u_1(u_3(f_Bv))$$

depending on whether B is relevant or not, and we obtain the extracted terms

$$\lambda\vec{x}, \vec{y}.e^{\text{ets}}_{C,D}(g^{\text{ets}}_B\vec{y}\vec{x}) \quad \text{or} \quad \lambda\vec{x}, \vec{y}.e^{\text{ets}}_{C,D}(\vec{x}\vec{y}).$$

(iii) Let $B \equiv \vec{C} \to R$. Since B is irrelevant we have $R \notin \mathcal{C}_L$. Then $(\vec{C} \to R)^A \equiv \vec{C}^A \to R$. We must derive

$$(\vec{C}^A \to R) \to \vec{C} \to R.$$

But this is easy, using the IH $d_{C_i}\colon C_i \to C_i^A$. Clearly $f^{\text{ets}}_B \equiv \varepsilon$.

(iv) *Case* R with $R \in \mathcal{C}_L$, $R \not\equiv \bot$. Then $R^A \equiv (R \to A) \to A$, and we must derive

$$((R \to A) \to A) \to (R \to A) \to A,$$

which is trivial. We have $g_R^{\mathrm{ets}} \equiv \lambda\vec{x}\,\vec{x}$.

Case R with $R \equiv \bot$. Then $R^A \equiv A$, and we must derive

$$A \to (\bot \to A) \to A,$$

which again is trivial. We have $g_\bot^{\mathrm{ets}} \equiv \lambda\vec{x}, \vec{y}\,\vec{x}$.

Case $C \to B$. By assumption B is relevant. We must derive

$$(C^A \to B^A) \to ((C \to B) \to A) \to A.$$

So assume $u\colon C^A \to B^A$ and $v\colon (C \to B) \to A$.

We first consider the case where C is relevant. Then we have $e_{C,B}\colon ((C \to B) \to A) \to C^A$ by IH(ii), hence B^A (using v and u). By IH(iv) for the shorter formula B we know $g_B\colon B^A \to (B \to A) \to A$ and hence $(B \to A) \to A$. But $B \to A$ can easily be derived from our hypothesis $v\colon (C \to B) \to A$ and hence we obtain A, as required. The derivation term is

$$g_{C \to B} \equiv \lambda u, v.g_B(u(e_{C,B}v))\lambda u_1^B.v\lambda u_2^C u_1$$

and we get

$$g_{C\to B}^{\mathrm{ets}} \equiv \lambda\vec{x}, \vec{y}.g_B^{\mathrm{ets}}(\vec{x}(e_{C,B}^{\mathrm{ets}}\vec{y}))\vec{y}.$$

We finally consider the case where C is irrelevant. The derivation now uses Cases_C; so it suffices to first derive A under the additional hypothesis C, and then derive A under the additional hypothesis $\neg C$.

So assume $u^+\colon C$. Since by IH(i) $d_C\colon C \to C^A$ we obtain C^A and hence B^A (using $u\colon C^A \to B^A$). By IH(iv) for the shorter formula B we know $g_B\colon B^A \to (B \to A) \to A$ and hence $(B \to A) \to A$. But $B \to A$ can easily be derived from our hypothesis $v\colon (C \to B) \to A$ and hence we obtain A, as required.

Now assume $u^-\colon \neg C$. Then by ex–falso–quodlibet we obtain $C \to B$ and hence A, using our hypothesis $v\colon (C \to B) \to A$.

The derivation term is

$$g_{C\to B} \equiv \lambda u, v.\mathrm{Cases}_C(\lambda u^+.g_B(u(d_C u^+))\lambda u_1^B.v\lambda u_2^C u_1)(\lambda u^-.v\lambda u_3^C.\mathrm{Efq}_B(u^- u_3))$$

and we get

$$g_{C\to B}^{\mathrm{ets}} \equiv \lambda\vec{x}, \vec{y}.\ \mathbf{if}\quad C\quad \mathbf{then}\quad g_B^{\mathrm{ets}}(\vec{x}d_C^{\mathrm{ets}})\vec{y}\quad \mathbf{else}\quad \vec{y}\quad \mathbf{fi}. \qquad \square$$

Lemma 2*. *Let B_i be quantifier–free formulas and $A := \exists^*\vec{y} \bigwedge\!\!\!\bigwedge_i B_i$. Then we can find a derivation of*

$$\forall\vec{y}.B_1^A \to \ldots \to B_m^A \to A$$

involving Cases_{B_i} *and* $d_{C_{ij}}\colon C_{ij} \to C_{ij}^A$ *for relevant* $B_i \equiv \vec{C}_i \to P_i$ *and* $f_{B_j}\colon B_j^A \to B_j$ *for irrelevant* B_j.

Proof. Let $\vec{y}$ be given and assume $v_1\colon \vec{B}_1^A, \dots, v_m\colon \vec{B}_m^A$. We must show A.

We may assume that B_i for $i \in \{1, \dots, k\}$ be relevant and B_j for $j \in \{k+1, \dots, m\}$ be irrelevant.

Case $u_i^-\colon \neg B_i$ for some $i \in \{1, \dots, k\}$. Let $B_i \equiv \vec{C}_i \to P_i$ with atoms P_i. Then

$$v_i\colon \vec{C}_i^{\,A} \to (P_i \to A) \to A$$

in case $P_i \not\equiv \bot$ and

$$v_i\colon \vec{C}_i^{\,A} \to A$$

in case $P_i \equiv \bot$. We have

$$\begin{aligned} e_{ij}[u_i^-]&\colon\equiv \mathrm{Stab}_{C_{ij}} \lambda v^{\neg C_{ij}}.u_i^- \lambda \vec{u}^{\vec{C}_i}.\mathrm{Efq}_{P_i}(vu_j)\colon C_{ij}, \\ e_i[u_i^-]&\colon\equiv \lambda w_i^{P_i}.u_i^- \lambda \vec{u}^{\vec{C}_i} w_i\colon \neg P_i. \end{aligned}$$

Using $d_{C_{ij}}\colon C_{ij} \to C_{ij}^A$ from Lemma 1* we obtain in case $P_i \not\equiv \bot$

$$v_i(d_{C_{i1}} e_{i1}) \dots (d_{C_{in_i}} e_{in_i})(\lambda w_i^{P_i}.\mathrm{Efq}_A(e_i w_i))\colon A$$

and in case $P_i \equiv \bot$

$$v_i(d_{C_{i1}} e_{i1}) \dots (d_{C_{in_i}} e_{in_i})\colon A.$$

Case $u_i^+\colon B_i$ for all $i \in \{1, \dots, k\}$. Then

$$\exists^+ \vec{y} \langle u_1^+, \dots, u_k^+, f_{B_{k+1}} v_{k+1}, \dots, f_{B_m} v_m \rangle\colon A$$

The extracted terms for this derivation are

$$\begin{aligned} d^{\mathrm{ets}} \equiv \lambda \vec{y}, \vec{x}_1, \dots, \vec{x}_m.\ &\mathbf{if} \quad \neg B_1 \quad \mathbf{then} \quad \vec{x}_1 d^{\mathrm{ets}}_{C_{11}} \dots d^{\mathrm{ets}}_{C_{1n_1}} \vec{0} \quad \mathbf{else} \\ &\dots \\ &\mathbf{if} \quad \neg B_k \quad \mathbf{then} \quad \vec{x}_k d^{\mathrm{ets}}_{C_{k1}} \dots d^{\mathrm{ets}}_{C_{kn_k}} \vec{0} \quad \mathbf{else} \\ &\vec{y} \quad \mathbf{fi} \dots \mathbf{fi}, \end{aligned}$$

where $\vec{x}_i$ is the list of variables associated with $v_i\colon B_i^A$. □

This A–translation relative to L simplifies the extracted terms a lot. First the derivations of $C \to C^A$ are necessary only for those lemmata $\forall C$ where C involves L–critical relation symbols (for otherwise we have $C^A \equiv C$). For the other lemmata a derivation of $C \to C^A$ provided by Lemma 1* involves in most cases only very few case distinctions. Finally, in the derivation given by Lemma 2* of the A–translation of our false assumption

$\bigwedge\!\!\!\bigwedge_i B_i^A \to A$ direct case distinctions are only necessary for the *relevant* B_i; for the other B_j we have a derivation of $B_j^A \to B_i$ by Lemma 1*.

Let us now come back to our initial example and study the effect of our refinements there. We can relativize the A–translation to the set L of formulas consisting of

$$\begin{aligned}
&n < f(0) \to \bot, \\
&n < f(g(n)), \\
&(n < f(m) \to \bot) \to n < f(m+1) \to \bot.
\end{aligned}$$

Since no positive subformula of an L–formula is an implication with a $<$–atom as its conclusion there are no L–critical relation symbols. Hence only negations $\vec{B} \to \bot$ are L–relevant.

We now repeat our treatment of the root example, based of the A–translation relative to L and the refined Lemmata 1* and 2*. The derivation term corresponding to the informal proof is

$$d := \mathrm{Ind}_{m,\neg n<f(m)} n(v_1 n)u(g(n))(v_2 n) \colon \bot.$$

Now let

$$A := \exists^* m. \neg n < f(m) \wedge n < f(m+1).$$

The program extracted from d is

$$(d^{\mathrm{tr}})^{\mathrm{ets}} \equiv (d^A)^{\mathrm{ets}}[d_u^{\mathrm{ets}}, d_{v_1}^{\mathrm{ets}}, d_{v_2}^{\mathrm{ets}}] \colon \mathrm{nat}.$$

$(d^A)^{\mathrm{ets}}$ is the same as d exept that $\mathrm{Ind}_{m,\neg n<f(m)}$ has to be replaced by

$$\lambda n\, R_{\mathrm{nat},\mathrm{nat}}$$

(since $\tau((n < f(m) \to \bot)^A) = \tau(n < f(m) \to A) = \mathrm{nat}$), and the assumption variables u, v_1 in d have to be replaced by (unary lists of) object variables

$$\begin{aligned}
&x_u \colon \mathrm{nat} \to \mathrm{nat} \to \mathrm{nat}, \\
&x_{v_1} \colon \mathrm{nat} \to \mathrm{nat},
\end{aligned}$$

whereas v_2 has to be replaced by the empty list. The subprograms $d_u^{\mathrm{ets}}, d_{v_1}^{\mathrm{ets}}$ (of the same types as x_u, x_{v_1}) are given by (cf. Lemmata 2* and 1*)

$$\begin{aligned}
d_u^{\mathrm{ets}} &\equiv \lambda m, x.\ \mathbf{if}\quad n < f(m) \quad \mathbf{then} \quad x \quad \mathbf{else} \\
&\qquad\qquad\quad m \quad \mathbf{fi}, \\
d_{v_1}^{\mathrm{ets}} &\equiv \lambda n\, 0.
\end{aligned}$$

Hence the normal form of $(d^{\mathrm{tr}})^{\mathrm{ets}}$ is

$$R0(\lambda m, x.\ \mathbf{if}\quad n < f(m) \quad \mathbf{then} \quad x \quad \mathbf{else} \quad m \quad \mathbf{fi})(g(n)).$$

Informally, $(d^{\mathrm{tr}})^{\mathrm{ets}} = h(g(n))$ where $h\colon \mathrm{nat} \to \mathrm{nat}$ is such that

$$h(0) = 0,$$
$$h(m+1) = \textbf{if} \quad n < f(m) \quad \textbf{then} \quad h(m) \quad \textbf{else} \quad m \quad \textbf{fi}.$$

Problem. *Let $C = C[\vec{y}]$ be the quantifier–free formula*

$$C[\vec{y}] \equiv (\vec{C}_1 \to P_1) \to \ldots \to (\vec{C}_m \to P_m) \to Q_1 \to \ldots \to Q_n \to R$$

with atoms P_i, Q_j, R. Assume that we have derivations $d_{C_{ik}}\colon C_{ik} \to C_{ik}^A$ with extracted terms $d_{C_{ik}}^{\mathrm{ets}}$. Show that there exists a derivation of $\forall \vec{y}.C \to C^A$ whose extracted termlist has the form

$$\begin{aligned}\lambda \vec{y}, \vec{x}_1, \ldots, \vec{x}_m, \vec{x}'_1, \ldots, \vec{x}'_n, \vec{z}.\ & \textbf{if} \quad \neg(\vec{C}_1 \to P_1) \quad \textbf{then} \quad \vec{x}_1 d_{C_{11}}^{\mathrm{ets}} \ldots d_{C_{1n_1}}^{\mathrm{ets}} \vec{0} \quad \textbf{else} \\ & \ldots \\ & \textbf{if} \quad \neg(\vec{C}_m \to P_m) \quad \textbf{then} \quad \vec{x}_m d_{C_{m1}}^{\mathrm{ets}} \ldots d_{C_{mn_m}}^{\mathrm{ets}} \vec{0} \quad \textbf{else} \\ & \vec{x}'_1(\vec{x}'_2 \ldots (\vec{x}'_n \vec{z}) \ldots) \quad \textbf{fi} \ldots \textbf{fi}.\end{aligned}$$

Solution. Assume

$$\begin{aligned} u &\colon (\vec{C}_1 \to P_1) \to \ldots \to (\vec{C}_m \to P_m) \to Q_1 \to \ldots \to Q_n \to R, \\ v_i &\colon \vec{C}_i^A \to (P_i \to A) \to A, \\ v'_j &\colon (Q_j \to A) \to A, \\ w &\colon R \to A. \end{aligned}$$

Assume further $d_{C_{ik}}\colon C_{ik} \to C_{ik}^A$.

Case $\vec{u}_i\colon \vec{C}_i$, $u'_i\colon \neg P_i$ for some i. Then

$$v_i(d_{C_{i1}} u_{i1}) \ldots (d_{C_{im_i}} u_{im_i})(\lambda w_i^{P_i}.\mathrm{Efq}_A(u'_i w_i))\colon A$$

Case $u''_1\colon \vec{C}_1 \to P_1, \ldots, u''_m\colon \vec{C}_m \to P_m$. Then we have $uu''_1 \ldots u''_m\colon Q_1 \to \ldots \to Q_n \to R$ and hence

$$v'_1(\lambda u_1^{Q_1}.v'_2(\lambda u_2^{Q_2}.\ldots.v'_n(\lambda u_n^{Q_n}.w(uu''_1 \ldots u''_m u_1 \ldots u_n))\ldots))\colon A.$$

For the extracted terms let $\vec{x}_i$ be the list of variables associated with v_i, $\vec{x}'_j$ be the list of variables associated with v'_j and $\vec{z}$ be the list of variables associated with w. Then clearly the extracted termlist is as given in the statement of the lemma. □

Part III: Classifying arithmetical proofs

10 Ordinals below epsilon zero

We want to discuss the derivability and underivability of initial cases of transfinite induction in arithmetical systems. In order to do that we shall need some knowledge and notations for ordinals. Now we do not want to assume set theory here; hence we introduce a certain initial segment of the ordinal (the ordinals $< \varepsilon_0$) in a formal, combinatorial way, i.e. via ordinal notations. Our treatment is based on the Cantor normal form for ordinals; cf. [1]. We also introduce some elementary relations and operations for such ordinal notations, which will be used later.

We define the two notions

- α is an ordinal notation
- $\alpha < \beta$ for ordinal notations α, β

simultaneously by induction:

1. If $\alpha_m, \dots, \alpha_0$ are ordinal notations and $\alpha_m \geq \dots \geq \alpha_0$ (where $\alpha \geq \beta$ means $\alpha > \beta$ or $\alpha = \beta$), then
$$\omega^{\alpha_m} + \dots + \omega^{\alpha_0}$$
is an ordinal notation. Note that the empty sum denoted by 0 is allowed here.
2. If $\omega^{\alpha_m} + \dots + \omega^{\alpha_0}$ and $\omega^{\beta_n} + \dots + \omega^{\beta_0}$ are ordinal notations, then
$$\omega^{\alpha_m} + \dots + \omega^{\alpha_0} < \omega^{\beta_n} + \dots + \omega^{\beta_0}$$
iff there is an $i \geq 0$ such that $\alpha_{m-i} < \beta_{n-i}$, $\alpha_{m-i+1} = \beta_{n-i+1}, \dots, \alpha_m = \beta_n$, or else $m < n$ and $\alpha_m = \beta_n, \dots, \alpha_0 = \beta_{n-m}$.

It is easy to see (by induction on the levels in the inductive definition) that $<$ is a linear order with 0 being the smallest element.

We shall use the notation 1 for ω^0, a for $\omega^0 + \dots + \omega^0$ with a copies of ω^0 and $\omega^\alpha a$ for $\omega^\alpha + \dots + \omega^\alpha$ again with a copies of ω^α.

We now define addition for ordinal notations:

$$\omega^{\alpha_m} + \dots + \omega^{\alpha_0} + \omega^{\beta_n} + \dots + \omega^{\beta_0} := \omega^{\alpha_m} + \dots + \omega^{\alpha_i} + \omega^{\beta_n} + \dots + \omega^{\beta_0}$$

where i is minimal such that $\alpha_i \geq \beta_n$.

It is easy to see that $+$ is an associative operation which is strictly monotonic in the second argument and weakly monotonic in the first argument. Note that $+$ is not commutative: $1 + \omega = \omega \neq \omega + 1$.

The *natural* (or Hessenberg) sum of two ordinal notations is defined by

$$(\omega^{\alpha_m} + \cdots + \omega^{\alpha_0}) \# (\omega^{\beta_n} + \cdots + \omega^{\beta_0}) := \omega^{\gamma_{m+n}} + \cdots + \omega^{\gamma_0},$$

where $\gamma_{m+n}, \ldots, \gamma_0$ is a decreasing permutation of $\alpha_m, \ldots, \alpha_0, \beta_n, \ldots, \beta_0$.

Again it is easy to see that $\#$ is associative, commutative and strictly monotonic in both arguments.

We will also need to know how ordinal notations of the form $\beta + \omega^\alpha$ can be approximated from below. First note that

$$\delta < \alpha \to \beta + \omega^\delta a < \beta + \omega^\alpha.$$

Furthermore, for any $\gamma < \beta + \omega^\alpha$ we can find a $\delta < \alpha$ and an a such that

$$\gamma < \beta + \omega^\delta a.$$

We now define 2^α for ordinal notations α. Let $\alpha_m \geq \cdots \alpha_0 \geq \omega > k_n \geq \cdots \geq k_1 > 0$. Then

$$2^{\omega^{\alpha_m} + \cdots + \omega^{\alpha_0} + \omega^{k_n} + \cdots + \omega^{k_1} + \omega^0 a} := \omega^{\omega^{\alpha_m} + \cdots + \omega^{\alpha_0} + \omega^{k_n - 1} + \cdots + \omega^{k_1 - 1}} 2^a.$$

It is easy to see that $2^{\alpha+1} = 2^\alpha + 2^\alpha$ and that 2^α is strictly monotonic in α.

In order to work with ordinal notations in a purely arithmetical system we set up a bijection between ordinal notations and nonnegative integers (i.e. a Gödel numbering). For its definition it is useful to refer to ordinal notations in the form

$$\omega^{\alpha_m} a_m + \cdots + \omega^{\alpha_0} a_0 \quad \text{with} \quad \alpha_m > \cdots > \alpha_0.$$

For any ordinal notation α we define its Gödel number $|\alpha|$ inductively by

$$|0| := 0,$$

$$|\omega^{\alpha_m} a_m + \cdots + \omega^{\alpha_0} a_0| := \Big(\prod_{i \leq m} p_{|\alpha_i|}^{a_i}\Big) - 1.$$

For any nonnegative integer x we define its corresponding ordinal notation $\mathrm{o}(x)$ inductively by

$$\mathrm{o}(0) = 0$$

$$\mathrm{o}\Big(\Big(\prod_{i \leq m} p_i^{a_i}\Big) - 1\Big) = \sum_{i \leq m} \omega^{\mathrm{o}(i)} a_i$$

where the sum is to be understood as the natural sum.

Lemma.

1. $\mathrm{o}(|\alpha|) = \alpha$,
2. $|\mathrm{o}(x)| = x$.

This can be proved easily by induction. □

Hence we have a bijection between ordinal notations and nonnegative integers. Using this bijection we can transfer our relations and operations on ordinal notations to computable relations and operations on nonnegative integers. We will use the notations

$$\begin{aligned} x \prec y &\quad \text{for} \quad \mathrm{o}(x) < \mathrm{o}(y), \\ \omega^x &\quad \text{for} \quad |\omega^{\mathrm{o}(x)}|, \\ x \oplus y &\quad \text{for} \quad |\mathrm{o}(x) + \mathrm{o}(y)|. \end{aligned}$$

11 Provability of initial cases of transfinite induction

We now derive initial cases of the principle of transfinite induction in arithmetic, i.e. of

$$[\forall x.(\forall y.y \prec x \to P(y)) \to P(x)] \to \forall x.x \prec a \to P(x)$$

for some number a and a predicate symbol P. In §13 we will see that our results here are optimal in the sense that for larger segments of the ordinals transfinite induction is underivable. All these results are due to Gentzen [12].

Our arithmetical systems are based on a fixed (possibly countably infinite) supply of function constants and predicate constants which are assumed to denote fixed functions and predicates on the nonnegative integers for which a computation procedure is known. An example is the formal system of arithmetic described in §7. Among the function constants there must be a constant S for the successor function and 0 for (the 0–place function) zero. Among the predicate constants there must be a constant $=$ for equality and $\prec$ for the ordering of type ε_0 of the natural numbers, as introduced in §10. In order to formulate the general principle of transfinite induction we also assume that a predicate symbol P is present.

Terms are built up from object variables x, y, z by means of $f(t_1, \dots, t_m)$, where f is a function constant. We identify closed terms which have the same value; this is a convenient way to express in our formal systems the assumption that for each function constant a computation procedure is known. Terms of the form $S(S(\dots S(0)\dots))$ are called *numbers*. We use the notation $S^i 0$ or even i for them. *Formulas* are built up from prime formulas $P(t_1, \dots, t_m)$ with P a predicate constant or a predicate symbol and $\perp$ by means of $(A \to B)$ and $\forall x\, A$. As usual we abbreviate $A \to \perp$ by $\neg A$.

The *axioms* of our arithmetical systems will always include the Peano–axioms

$$\forall x, y.S(x) = S(y) \to x = y,$$
$$\forall x\, S(x) \neq 0.$$

Any instance of the induction scheme

$$A[0/x] \to (\forall x.A \to A[S(x)/x]) \to \forall x\, A$$

with A an arbitrary formula is an axiom of full arithmetic Z. We will also consider subsystems Z_k of Z where the formulas A in the induction scheme are restricted to formulas of level $\text{lev}(A) \leq k$. The level of of a formula A is defined by

$$\text{lev}(P) := 0 \quad \text{for any atom } P,$$
$$\text{lev}(A \to B) := \max(\text{lev}(A) + 1, \text{lev}(B)),$$
$$\text{lev}(\forall x\, A) := \max(1, \text{lev}(A)).$$

In addition, in any arithmetical system we have the equality axioms

$$\begin{aligned}
&\forall x. x = x,\\
&\forall x, y. x = y \to y = x,\\
&\forall x, y, z. x = y \to y = z \to x = z,\\
&\forall \vec{x}, \vec{y}. x_1 = y_1 \to \ldots \to x_m = y_m \to f(\vec{x}) = f(\vec{y}),\\
&\forall \vec{x}, \vec{y}. x_1 = y_1 \to \ldots \to x_m = y_m \to P(\vec{x}) \to P(\vec{y})
\end{aligned}$$

for any function constant f and predicate constant or predicate symbol P. We also require irreflexivity and transitivity for $\prec$ as axioms. For our negative results we can also allow that for any predicate symbol P the stability axioms

$$\forall \vec{x}. \neg\neg P(\vec{x}) \to P(\vec{x})$$

are present. However, for the positive results no stability axioms are needed. We express our assumption that for any predicate constant a decision procedure is known by adding the axiom

$$P(S^{i_1}0, \ldots, S^{i_m}0)$$

whenever $P(\vec{i})$ is true, and

$$\neg P(S^{i_1}0, \ldots, S^{i_m}0)$$

whenever $P(\vec{i})$ is false.

We finally allow in any of our arithmetical systems Z an arbitrary supply of true Π–formulas as axioms. Our (positive and negative) results concerning initial cases of transfinite recursion will not depend on which of those axioms we have chosen, except that for the positive results we always assume the universal closures of

$$\begin{aligned}
&x \not\prec 0, && (1)\\
&z \prec y \oplus \omega^0 \to z \not\prec y \to z \neq y \to \bot, && (2)\\
&x \oplus 0 = x, && (3)\\
&x \oplus (y \oplus z) = (x \oplus y) \oplus z, && (4)\\
&0 \oplus x = x, && (5)\\
&\omega^x 0 = 0, && (6)\\
&\omega^x (Sy) = \omega^x y \oplus \omega^x, && (7)\\
&z \prec y \oplus \omega^x \to x \neq 0 \to z \prec y \oplus \omega^{f(x,y,z)} g(x, y, z), && (8)\\
&z \prec y \oplus \omega^x \to x \neq 0 \to f(x, y, z) \prec x, && (9)
\end{aligned}$$

where in (9) f and g are function constants.

Theorem 11.1 (Gentzen). *Transfinite induction up to ω_n (with $\omega_1 := \omega, \omega_{n+1} := \omega^{\omega_n}$), i.e. the formula*

$$\forall x \left((\forall y. y \prec x \to A[y]) \to A \right) \to \forall x. x \prec \omega_n \to A$$

is derivable in Z.

Proof. To any formula A we assign a formula A^+ (with respect to a fixed variable x) by

$$A^+ := \forall y.(\forall z.z \prec y \to A[z/x]) \to \forall z.z \prec y \oplus \omega^x \to A[z/x].$$

We first show

$$A \text{ is progressive} \implies A^+ \text{ is progressive},$$

where "B is *progressive*" means $\forall x.(\forall y.y \prec x \to B[y]) \to B$. So assume that A is progressive and

$$\forall y.y \prec x \to A^+[y]. \tag{10}$$

We have to show A^+. So assume further

$$\forall z.z \prec y \to A[z] \tag{11}$$

and $z \prec y \oplus \omega^x$. We have to show $A[z]$. Case $x = 0$. From $z \prec y \oplus \omega^0$ we have by (2) $z \prec y \vee z = y$. If $z \prec y$, then $A[z]$ follows from (11), and if $z = y$, then $A[z]$ follows from (11) and the progressiveness of A. Case $x \neq 0$. From $z \prec y \oplus \omega^x$ we obtain $z \prec y \oplus \omega^{f(x,y,z)} g(x,y,z)$ by (8) and $f(x,y,z) \prec x$ by (9). From (10) we obtain $A^+[f(x,y,z)]$. By the definition of A^+ we get

$$(\forall u.u \prec y \oplus \omega^{f(x,y,z)} v \to A[u]) \to \forall u.u \prec (y \oplus \omega^{f(x,y,z)} v) \oplus \omega^{f(x,y,z)} \to A[u]$$

and hence, using (4) and (7)

$$(\forall u.u \prec y \oplus \omega^{f(x,y,z)} v \to A[u]) \to \forall u.u \prec y \oplus \omega^{f(x,y,z)} S(v) \to A[u].$$

Also from (11) and (6), (3) we obtain

$$\forall u.u \prec y \oplus \omega^{f(x,y,z)} 0 \to A[u].$$

Using an appropriate instance of the induction scheme we can conclude

$$\forall u.u \prec y \oplus \omega^{f(x,y,z)} g(x,y,z) \to A[u]$$

and hence $A[z]$.

We now show, by induction on n, how to obtain a derivation of

$$\forall x\,((\forall y.y \prec x \to A[y]) \to A) \to \forall x.x \prec \omega_n \to A.$$

So assume the left–hand side, i.e. assume that A is progressive. *Case* 0. From $x \prec \omega_0$ we get $x = 0$ by (5), (2) and (1), and $A[0]$ follows from the progressiveness of A by (1). *Case* $n + 1$. Since A is progressive, by what we have shown above also A^+ is progressive. Applying the IH to A^+ yields $\forall x.x \prec \omega_n \to A^+$, and hence $A^+[\omega_n]$ by the progressiveness of A^+. Now the definition of A^+ (together with (1) and (5)) yields $\forall z.z \prec \omega^{\omega_n} \to A[z]$. □

Note that in these derivations the induction scheme was used for formulas of unbounded complexity.

We now want to refine the Theorem to a corresponding result for the subsystems Z_k of Z. Note first that if A is a formula of level $\leq k$, then the formula A^+ constructed in the proof of the Theorem has level $\leq k+1$, and for the proof of

$$A \text{ is progressive} \implies A^+ \text{ is progressive}$$

we have used induction with an induction formula of level $\leq k$.

Now let A be of level ≤ 1, and assume that A is progressive. Let $A^0 := A, A^{i+1} := (A^i)^+$. Then A^i is of level $\leq i+1$, and hence in Z_k we can derive that $A^1, A^2, \ldots A^k$ are all progressive. Let $\omega_1[m] := m$, $\omega_{i+1}[m] = \omega^{\omega_i[m]}$. Since in Z_k we can derive that A^k is progressive, we can also derive $A^k[0], A^k[1], A^k[2]$ and generally $A^k[m]$ for any m, i.e. $A^k[\omega_1[m]]$. But since

$$A^k \equiv (A^{k-1})^+ \equiv \forall y.(\forall z.z \prec y \to A^{k-1}[z]) \to \forall z.z \prec y \oplus \omega^x \to A^{k-1}[z],$$

we first get (with $y=0$) $\forall z.z \prec \omega_2[m] \to A^{k-1}[z]$ and then $A^{k-1}[\omega_2[m]]$ by the progressiveness of A^{k-1}. Repeating this argument we finally obtain $\forall z.z \prec \omega_{k+1}[m] \to A^0[z]$. Hence we have

Theorem 11.2. *Let A be a formula of level 1. Then in Z_k we can derive transfinite induction for A up to $\omega_{k+1}[m]$ for any m, i.e.*

$$Z_k \vdash [\forall x.(\forall y.y \prec x \to A[y]) \to A] \to \forall x.x \prec \omega_{k+1}[m] \to A. \qquad \square$$

If more generally we start out with a formula A of level $\leq \ell$ instead, where $1 \leq \ell \leq k$, then a similar argument yields the following result (cf. Parsons [17]).

Theorem 11.3. *Let A be a formula of level $\leq \ell$, $1 \leq \ell \leq k$. Then in Z_k we can derive transfinite induction for A up to $\omega_{k+2-\ell}[m]$ for any m, i.e.*

$$Z_k \vdash [\forall x.(\forall y.y \prec x \to A[y]) \to A] \to \forall x.x \prec \omega_{k+2-\ell}[m] \to A. \qquad \square$$

Our next aim is to prove that these bounds are sharp. More precisely, we will show that in Z (no matter how many true Π–formulas we have added as axioms) one cannot derive transfinite induction up to ε_0, i.e. the formula

$$[\forall x.(\forall y.y \prec x \to P(y)) \to P(x)] \to \forall x\, P(x)$$

with a free predicate symbol P, and that in Z_k one cannot derive transfinite induction up to ω_{k+1}, i.e. the formula

$$[\forall x.(\forall y.y \prec x \to P(y)) \to P(x)] \to \forall x.x \prec \omega_{k+1} \to P(x).$$

This will follow from the method of normalization applied to arithmetical systems, which we have to develop first.

12 Normalization for arithmetic with the omega rule

We will show in §14 that a normalization theorem does not hold for a system of arithmetic like Z in §11, in the sense that for any formula A derivable in Z there is a derivation of the same formula A in Z which only uses formulas of a level bounded by the level of A. The reason for this failure is the presence of the induction axioms, which can be of arbitrary level.

Here we remove that obstacle against normalization and replace the induction axioms by a rule with infinitely many premises, the so–called ω–rule (suggested by Hilbert and studied by Lorenzen, Novikov and Schütte), which allows to conclude $\forall x\, A$ from $A[0], A[1], A[2], \cdots$.

Clearly this ω–rule can also be used to replace the rule $\forall^+$. As a consequence we do not need to consider free object variables.

So we introduce the system Z^∞ of ω–arithmetic as follows. Z^∞ has the same language and — apart from the induction axioms — the same axioms as Z. Derivations in Z^∞ are infinite objects; they are built up from assumption variables u^A, v^B and constants ax^A for any axiom A of Z other than an induction axiom by means of the rules

$$
\begin{aligned}
&(\lambda u^A d^B)^{A\to B},\\
&(d^{A\to B} e^A)^B,\\
&\langle d_i^{A[i]}\rangle_{i<\omega}^{\forall x\, A},\\
&(d^{\forall x\, A} i)^{A[i]}
\end{aligned}
$$

denoted by $\to^+, \to^-, \omega$ and $\forall^-$, respectively.

More precisely, we define the notion of a $\vec{u}$–derivation (i.e. a derivation in Z^∞ with free assumption variables among $\vec{u}$) of height $\leq \alpha$ and degree $\leq k$ inductively, as below.

Note that derivations are infinite objects now. They may be viewed as mappings from finite lists of natural numbers (= nodes in the derivation tree) to lists of data including the formula appearing at that node, the rule applied last, a list of assumption variables including all those free in the subderivation (starting at that node), a bound on the height of the subderivation, and a bound on the degree of the subderivation.

Intuitively, the degree of a derivation is the least number $\geq$ the level of any subderivation $\lambda x\, d$ in a context $(\lambda x\, d)e$ or $\langle d_i\rangle_{i<\omega}$ in a context $\langle d_i\rangle_{i<\omega} j$, where the level of a derivation is the level of its type, i.e. the formula it derives. This notion of a degree is needed for the normalization proof we give below.

* Any assumption variable u^A and any axiom ax^A is a $\vec{u}$–derivation of height $\leq \alpha$ and degree $\leq k$, for any list $\vec{u}$ of assumption variables (containing u in the first case), ordinal α and number k.

$\to^+$ If d^B is a $\vec{u}, u, \vec{v}$–derivation of height $\le \alpha_0 < \alpha$ and degree $\le k$, then $(\lambda u^A d^B)^{A\to B}$ is a $\vec{u}, \vec{v}$–derivation of height $\le \alpha$ and degree $\le k$.

$\to^-$ If $d^{A\to B}$ and e^A are $\vec{u}$–derivations of heights $\le \alpha_i < \alpha$ and degrees $\le k_i \le k$ $(i = 1, 2)$, then $(d^{A\to B} e^A)^B$ is a $\vec{u}$–derivation of height $\le \alpha$ and degree $\le m$ with $m = \max(k, \mathrm{lev}(A \to B))$, if $d^{A\to B}$ is generated by the rule $\to^+$, or of degree $\le k$ otherwise.

ω If $r_i^{A[i]}$ are $\vec{u}$–derivations of heights $\le \alpha_i < \alpha$ and degrees $\le k_i \le k$ $(i < \omega)$, then $\langle r_i^{A[i]} \rangle_{i<\omega}^{\forall x\, A}$ is an $\vec{u}$–derivation of height $\le \alpha$ and degree $\le k$.

$\forall^-$ If $d^{\forall x\, A}$ is a $\vec{u}$–derivation of height $\le \alpha_0 < \alpha$ and degree $\le k$, then $(d^{\forall x\, A} i)^{A[i]}$ is a $\vec{u}$–derivation of height $\le \alpha$ and degree $\le m$ with $m = \max(k, \text{level } \forall x\, A)$, if $d^{\forall x\, A}$ is generated by the rule ω, or of degree $\le k$ otherwise.

We now embed our systems Z_k (i.e. arithmetic with induction restricted to formulas of level $\le k$) and hence Z into Z^∞.

Lemma 12.1. *Let d^B be a derivation in Z_k with free assumption variables among $\vec{u}^{\vec{A}}$ which contains $\le m$ instances of the induction scheme all with induction formulas of level $\le k$. Let σ be a substitution of numbers for object variables such that $\vec{A}\sigma, B\sigma$ do not contain free object variables. Then we can find a $\vec{u}^{\vec{A}\sigma}$–derivation $(d_\infty)^{B\sigma}$ in Z^∞ of height $\le \omega^m + h$ for some $h < \omega$ and degree $\le k$.*

Proof by induction on the height of the given derivation, which we may assume to be in long normal form. The only case which requires some argument is when the derivation consists of two applications of $\to^-$ to an instance of the induction scheme. Then it must have the form

$$\mathrm{Ind}_{x,A} d^{A[0]} (\lambda x, v^A\, e^{A[S(x)]}).$$

By IH we obtain derivations

$$\begin{array}{rll} d_\infty^{A[0]} & \text{of height} & \le \omega^{m-1} + h_0 \\ e_\infty^{A[1]}[d_\infty^{A[0]}] & \text{of height} & \le \omega^{m-1} \cdot 2 + h_1, \\ e_\infty^{A[2]}[e_\infty^{A[1]}[e_\infty^{A[0]}]] & \text{of height} & \le \omega^{m-1} \cdot 3 + h_2 \end{array}$$

and so on, all of degree $\le k$. Combining all these derivations of $A[i]$ as premises of the ω–rule yields a derivation of $\forall x\, A$ of height $\le \omega^m$ and degree $\le k$. □

A derivation is called *convertible* if it is of the form $(\lambda u\, d)e$ or else $\langle d_i \rangle_{i<\omega} j$, which can be converted into $d[e/u]$ or d_j, respectively. Here $d[e/u]$ is obtained from d by substituting e for all free occurences of u in d. A derivation is called *normal* if it does not contain a convertible subderivation. Note that a derivation of degree 0 must be normal.

We want to define an operation which by repeated conversions transforms a given derivation into a normal one with the same end formula and no more assumption variables. The methods employed in §4 to achieve such a task have to be adapted properly in order to deal with the new situation of infinitary derivations. Here we give a particularly simple argument due to Tait [29].

Lemma 12.2. *If d is a $\vec{u}, u^A, \vec{v}$–derivation of height $\leq \alpha$ and degree $\leq k$ and e^A is a $\vec{u}, \vec{v}$–derivation of height $\leq \beta$ and degree $\leq \ell$, then $d[e/u]$ is an $\vec{u}, \vec{v}$–derivation of height $\leq \beta + \alpha$ and degree $\leq \max(k, \ell, \mathrm{lev}(e))$.*

This is proved by a straightforward induction on the height of d. □

Lemma 12.3. *For any $\vec{u}$–derivation d^A of height $\leq \alpha$ and degree $\leq k+1$ we can find a $\vec{u}$–derivation $(d^k)^A$ of height $\leq 2^\alpha$ and degree $\leq k$.*

Proof by induction on α. The only case which requires some argument is when the derivation is of the form de with d of height $\leq \alpha_1 < \alpha$ and e of height $\leq \alpha_2 < \alpha$. We first consider the subcase where $d^k = \lambda u\, d_1$ and $\mathrm{lev}(d) = k+1$. Then $\mathrm{lev}(e) \leq k$ by the definition of level, and hence $(d_1)[e^k/u]$ has degree $\leq k$ by Lemma 12.2. Furthermore, also by Lemma 12.2, $(d_1)[e^k/u]$ has height $\leq 2^{\alpha_2} + 2^{\alpha_1} \leq 2^{\max(\alpha_2, \alpha_1)+1} \leq 2^\alpha$. Hence we can take $(de)^k$ to be $(d_1)[e^k/u]$. If we are not in the above subcase, we can simply take $(de)^k$ to be $d^k e^k$. This derivation clearly has height $\leq 2^\alpha$. Also it has degree $\leq k$, which can be seen as follows. If $\mathrm{lev}(d) \leq k$ we are done. If however $\mathrm{lev}(d) \geq k+2$, then d must be of the form $d_0 d_1 \dots d_m$ for some assumption variable or axiom d_0 (since the given derivation has degree $\leq k+1$). But then d^k has the form $d_0 d_1^k \dots d_m^k$ and we are done again. (To be completely precise, this last statement has to be added to the formulation of the Lemma above and proved simultaneously with it). □

As an immediate consequence we obtain

Theorem 12.4 (Normalization for Z^∞). *For any $\vec{u}$–derivation d^A of height $\leq \alpha$ and degree $\leq k$ we can find a normal $\vec{u}$–derivation $(d^*)^A$ of height $\leq 2_k \alpha$ (where $2_0\alpha = \alpha$, $2_{m+1}\alpha = 2^{2^\alpha_m}$).* □

13 Unprovable initial cases of transfinite induction

We now apply the technique of normalization for arithmetic with the ω–rule for a proof that transfinite induction up to ε_0 is underivable in Z, i.e. of

$$Z \nvdash [\forall x.(\forall y.y \prec x \to P(y)) \to P(x)] \to \forall x\, P(x)$$

with a predicate symbol P, and that transfinite induction up to ω_{k+1} is underivable in Z_k, i.e. of

$$Z_k \nvdash [\forall x.(\forall y.y \prec x \to P(y)) \to P(x)] \to \forall x.x \prec \omega_{k+1} \to P(x).$$

Our proof is based on an idea of Schütte, which consists in adding a so–called *progression rule* to the infinitary systems. This rule allows to conclude $P(j)$ (where j is any number) from all $P(i)$ for $i \prec j$.

More precisely, we define the notion of a $\vec{u}$–derivation in $Z^\infty +$ Prog(P) of height $\leq \alpha$ and degree $\leq k$ by the inductive clauses of §12 and the additional clause Prog(P):

If $d_i^{P(i)}$ are $\vec{u}$–derivations of heights $\leq \alpha_i < \alpha$ and degrees $\leq k_i \leq k$ $(i \prec j)$, then $\langle d_i^{P(i)} \rangle_{i \prec j}^{P(j)}$ is an $\vec{u}$–derivation of height $\leq \alpha$ and degree $\leq k$.

Since this progression rule only deals with derivations of prime formulas it does not affect the degrees of derivations. Hence the proof of normalization for Z^∞ carries over unchanged to $Z^\infty + \text{Prog}(P)$. In particular we have

Lemma 13.1. *For any $\vec{u}$–derivation d^A in $Z^\infty + \text{Prog}(P)$ of height $\leq \alpha$ and degree $\leq k+1$ we can find an $\vec{u}$–derivation $(d^k)^A$ in $Z^\infty + \text{Prog}(P)$ of height $\leq 2^\alpha$ and degree $\leq k$.* □

We now show that from the progression rule for P we can easily derive the progressiveness of P.

Lemma 13.2. *We have a normal derivation of $\forall x.(\forall y.y \prec x \to P(y)) \to P(x)$ in $Z^\infty +$ Prog(P) with height ≤ 5.*

Proof. By the ω–rule it suffices to derive $(\forall y.y \prec j \to P(y)) \to P(j)$ for any j with height ≤ 4. We argue informally. Assume $\forall y.y \prec j \to P(y)$. By $\forall^-$ we have $i \prec j \to P(i)$ for any i. Now for any $i \prec j$ we have $i \prec j$ as an axiom; hence $P(i)$ for any such i. An application of the progression rule yields $P(j)$, with a derivation of height ≤ 3. Now by $\to^+$ and ω the claim follows. □

The crucial observation now is that a normal derivation of $P(|\beta|)$ must essentially have a height of at least β. However, to obtain the right estimates for our subsystems Z_k we cannot apply Lemma 13.1 down to degree 0 (i.e. to the normal form) but must stop already at degree 1. Such derivations, i.e. those of degree ≤ 1, will be called *almost normal*; they can also be analyzed easily. An almost normal derivation d in $Z^\infty + \mathrm{Prog}(P)$ is called a $P(|\vec{\alpha}|), \neg P(|\vec{\beta}|)$*–refutation* if d derives a formula $\vec{A} \to B$ with $\vec{A}$ and the free assumptions in d among $P(|\vec{\alpha}|) := P(|\alpha_1|), \dots, P(|\alpha_m|)$ and $\neg P(|\vec{\beta}|) := \neg P(|\beta_1|), \dots, \neg P(|\beta_n|)$ and true prime formulas, and B a false prime formula or else among $P(|\vec{\beta}|)$.

Lemma 13.3. *Let d be an almost normal $P(|\vec{\alpha}|), \neg P(|\vec{\beta}|)$–refutation of height $\leq |d|$ with $\vec{\alpha}$ and $\vec{\beta}$ disjoint. Then*

$$\min(\vec{\beta}) \leq |d| + \mathrm{lg}\vec{\alpha},$$

where $\mathrm{lg}\vec{\alpha}$ *denotes the length of the list* $\vec{\alpha}$*.*

Proof by induction on $|d|$. Note that we may assume that d does not contain the ω–rule, since any application of it must be in a context $\langle d_i \rangle_j$, which can be replaced by d_j. We can also assume that d contains $\forall^-$ only in a context where leading universal quantifiers of an axiom are removed. Note also that d cannot derive an instance $|\gamma| = |\delta| \to P(|\gamma|) \to P(|\delta|)$ of an equality axiom with $\gamma = \delta$ true, since we have assumed that $\vec{\alpha}$ and $\vec{\beta}$ are disjoint. We distinguish cases according to the last rule in d.

Case $\to^+$. By our definition of refutations the claim follows immediately from the IH.

Case $\to^-$. Then $d \equiv f^{A \to (\vec{A} \to B)} e^A$. If A is a true prime formula, the claim follows from the IH for f. If A is a false prime formula, the claim follows from the IH for e. If A is $\neg\neg P(|\gamma|)$ (and hence $f \equiv \mathrm{Stab}_P^{\forall x. \neg\neg P(x) \to P(x)} |\gamma|)$, then since the level of $\neg\neg P(|\gamma|)$ is 2 the derivation $e^{\neg\neg P(|\gamma|)}$ must end with an introduction rule, i.e. $s \equiv \lambda u^{\neg P(|\gamma|)} e_0^{\perp}$ (for otherwise, since no axiom contains some $\neg\neg P(d_0)$ as a strictly positive subformula, we would get a contradiction against the assumption that d has degree ≤ 1). The claim now follows from the IH for e_0. The only remaining case is when A is $P(|\gamma|)$. Then f is an almost normal $P(|\gamma|), P(|\vec{\alpha}|), \neg P(|\vec{\beta}|)$ –refutation and e is an almost normal $P(|\vec{\alpha}|), \neg P(|\vec{\beta}|), \neg P(|\gamma|)$ –refutation. We may assume that γ is not among $\vec{\alpha}$, since otherwise the claim follows immediately from the IH for f. Hence we have by the IH for f

$$\min(\vec{\beta}) \leq |f| + \mathrm{lg}\vec{\alpha} + 1 \leq |d| + \mathrm{lg}\vec{\alpha}.$$

Case $\mathrm{Prog}(P)$. Then $d \equiv \langle d_\delta^{P(|\delta|)} \rangle_{\delta < \gamma}^{P(|\gamma|)}$. By IH, since d_δ is a $P(|\vec{\alpha}|)$, $\neg P(|\vec{\beta}|)$, $\neg P(|\delta|)$ –refutation, we have for all $\delta < \gamma$

$$\min(\vec{\beta}, \delta) \leq |d_\delta| + \mathrm{lg}\vec{\alpha} < |d| + \mathrm{lg}\vec{\alpha}$$

and hence

$$\min(\vec{\beta}, \gamma) \leq |d| + \mathrm{lg}\vec{\alpha}. \qquad \square$$

Now we can show the following result (cf. Mints [13] and Parsons [17]).

Theorem 13.4. *Transfinite induction up to ε_0 is underivable in Z, i.e.*

$$Z \nvdash [\forall x.(\forall y.y \prec x \to P(y)) \to P(x)] \to \forall x\, P(x)$$

with a predicate symbol P, and transfinite induction up to ω_{k+1} is underivable in Z_k, i.e.

$$Z_k \nvdash [\forall x.(\forall y.y \prec x \to P(y)) \to P(x)] \to \forall x.x \prec \omega_{k+1} \to P(x).$$

Proof. We restrict ourselves to the second part. So assume that transfinite induction up to ω_{k+1} is derivable in Z_k. Then by the embedding of Z_k into Z^∞ (Lemma 12.1) and the normal derivability of the progressiveness of P in $Z^\infty + \mathrm{Prog}(P)$ with finite height (Lemma 13.2) we can conclude that $\forall x.x \prec \omega_{k+1} \to P(x)$ is derivable in $Z^\infty + \mathrm{Prog}(P)$ with height $< \omega^m + h$ for some $m, h < \omega$ and degree $\leq k$. Now $k-1$ applications of Lemma 13.1 yield a derivation of the same formula $\forall x.x \prec \omega_{k+1} \to P(x)$ in $Z^\infty + \mathrm{Prog}(P)$ with height $\leq \gamma = 2_{k-1}(\omega^m + h) < \omega_{k+1}$ and degree ≤ 1, hence also a derivation of $P(|\gamma + 1|)$ in $Z^\infty + \mathrm{Prog}(P)$ with height $\leq \gamma$ and degree ≤ 1. But this contradicts Lemma 13.3. $\square$

14 Normalization for arithmetic is impossible

The normalization theorem for first–order logic applied to arithmetic Z is not particularly useful since we may have used in our derivation induction axioms of arbitrary complexity. Hence it is tempting to first eliminate the induction scheme in favour of an induction rule allowing to conclude $\forall x\, A$ from a derivation of $A[0]$ and a derivation of $A[S(x)]$ with an additional assumption A to be cancelled at this point (note that this rule is equivalent to the induction scheme), and then to try to normalize the resulting derivation in the new system Z with the induction rule. We will apply our results from §13 to show that even a very weak form of the normalization theorem cannot hold in Z with the induction rule.

Theorem 14.1. *The following weak form of a normalization theorem for Z with the induction rule is false: For any $\vec{u}^{\vec{A}}$–derivation d^B with $\vec{A}, B$ formulas of degree $\leq \ell$ there is an $\vec{u}^{\vec{A}}$–derivation $(d^*)^B$ containing only formulas of degree $\leq k$, with k depending only on ℓ.*

Proof. Assume that such a normalization theorem would hold. Consider the formula

$$[\forall x.(\forall y.y \prec x \to P(y)) \to P(x)] \to \forall x.x \prec \omega_{n+1} \to P(x)$$

expressing transfinite induction up to ω_{n+1}, which is of degree 3. By Theorem 11.1 it is derivable in Z. Hence there exists a derivation of the same formula containing only formulas of degree $\leq k$, for some k independent of n. Hence Z_k derives transfinite induction up to ω_{n+1} for any n. But this clearly contradicts Theorem 13.1. $\square$

Appendix

15 Permutative conversions

Prawitz in [19] proves strong normalization for conversion rules on proof trees for logic with $\vee$ and $\exists$. The conversion rules not only include the usual β–rules but also so–called permutative rules. The method Prawitz uses is based on so-called strong validity predicates; they are a variant of Tait's notion of a strongly computable term treated in §4.

The following exposition is based on Prawitz' proof and on a study of it by van de Pol [32]. It uses derivation terms instead of proof trees for a cleaner exposition. Also the definition of an end segment is formalized. In the definition of strong validity a small oversight in Prawitz' formulation has been corrected.

We first define the reduction rules. In order to have a uniform notation it is convenient to write

$$\begin{array}{rll}
\supset^+(u,d) & \text{instead of} & (\lambda u^A\, d^B)^{A\to B}, \\
\supset^-(d,e) & \text{instead of} & (d^{A\to B} e^A)^B, \\
\wedge^+(d,e) & \text{instead of} & \langle d^A, e^B\rangle^{A\wedge B}, \\
\wedge_i^-(d) & \text{instead of} & \pi_i(d^{A_0\wedge A_1})^{A_i}, \\
\forall^+(x,d) & \text{instead of} & (\lambda x^\rho\, d^A)^{\forall x^\rho\, A}, \\
\forall^-(d,t) & \text{instead of} & (d^{\forall x^\rho\, A} t^\rho)^{A[t/x^\rho]}.
\end{array}$$

The conversion rules are

$$\begin{array}{l}
\supset^-(\supset^+(u,d),e) \to_0 d[e/u], \\
\wedge_i^-(\wedge^+(d_0,d_1)) \to_0 d_i, \\
\vee^-(\vee_i^+(d),u_0,e_0,u_1,e_1) \to_0 e_i[d/u_i], \\
\forall^-(\forall^+(x,d),t) \to_0 d[t/x], \\
\exists^-(\exists^+(t,d),x,u,e) \to_0 e[t,d/x,u], \\
\Box^-(\vee^-(d,u,e,v,f),\vec{\alpha}) \to_0 \vee^-(d,u,\Box^-(e,\vec{\alpha}),v,\Box^-(f,\vec{\alpha})), \\
\Box^-(\exists^-(d,x,u,e),\vec{\alpha}) \to_0 \exists^-(d,x,u,\Box^-(e,\vec{\alpha})).
\end{array}$$

Here we have used $\Box$ to denote an arbitrary element of $\{\supset,\wedge,\vee,\forall,\exists\}$. $\vec{\alpha}$ denotes an appropriate sequence of object– and assumption variables. For simplicity we write $\vee,\exists$ instead of $\vee^*,\exists^*$.

The right hand side of each rule is called an *immediate reduct* of the left hand side. In the last two cases we call it a *permutative reduct.*. From $\to_0$ one derives a one–step reduction relation $\to$ as usual (cf. §4).

We now show that $\to$ is terminating, i.e. that any reduction sequence starting with d terminates after finitely many steps. We write $d \to^* d'$ (or $d \to^+ d'$) if d' is a member of a reduction sequence (a reduction sequence with at least two elements) starting with d. Hence $\to^*$ is the reflexive transitive closure of $\to$.

Clearly conversion is compatible with substitution:

Lemma 1. *If $d \to d'$, then $d[t] \to d'[t]$ and $d[e] \to d'[e]$*

Proof by induction on the definition of $d \to d'$. □

We now define what it means that a derivation e is an *end segment* of the derivation d. This is written $\mathrm{ES}(d, e)$, and the relation ES is defined inductively as follows.

(i) $\mathrm{ES}(d, d)$.

(ii) If $\mathrm{ES}(d', e)$, then $\mathrm{ES}(\exists^-(d, x, u, d'), e)$.

(iii) If $\mathrm{ES}(d_i, e)$ for some i, then $\mathrm{ES}(\vee^-(d, u_0, d_0, u_1, d_1), e)$.

Clearly the end segment relation is transitive, i.e. $\mathrm{ES}(d, e)$ and $\mathrm{ES}(e, f)$ imply $\mathrm{ES}(d, f)$. We now prove that an introduction in the end segment of the main premise of an exists–elimination can always be removed by conversion.

Lemma 2. *Consider $\exists^-(d_1, x, u, f)$ and assume $\mathrm{ES}(d_1, \exists^+(t, e))$. Then there exists a d_2 such that $\exists^-(d_1, x, u, f) \to^+ d_2$ and $\mathrm{ES}(d_2, f[t, e/x, u])$.*

Proof by induction on the definition of $\mathrm{ES}(d_1, \exists^+(t, e))$.

Case $d_1 = \exists^+(t, e)$. Then we can carry out a proper conversion, i.e. let $d_2 = f[t, e/x, u]$.

Case $d_1 = \exists^-(g, y, v, d')$ with $\mathrm{ES}(d', \exists^+(t, e))$. Then we have an existential permutative conversion

$$\exists^-(d_1, x, u, f) \to_0 \exists^-(g, y, v, \exists^-(d', x, u, f)).$$

By IH we can reduce the last term to $\exists^-(g, y, v, d_2')$ with $\mathrm{ES}(d_2', f[t, e/x, u])$. Let $d_2 = \exists^-(g, y, v, d_2')$.

Case $d_1 = \vee^-(d, u_0, d_0, u_1, d_1)$ with $\mathrm{ES}(d_i, \exists^+(t, e))$. Similar. □

Similarly we can prove that an introduction in the end segment of an or–elimination can always be removed by conversion.

Lemma 3. *Consider $\vee^-(d_1, u_0, e_0, u_1, e_1)$ and assume $\mathrm{ES}(d_1, \vee_i^+(e))$. Then there exists a d_2 such that $\vee^-(d_1, u_0, e_0, u_1, e_1) \to^+ d_2$ and $\mathrm{ES}(d_2, e_i[e/u_i])$.* □

We now define the central notion of a *strongly valid derivation* d^A. The definition is by induction on the number of logical symbols in A, and for fixed A it is an inductive definition. We write $\mathrm{SV}(d)$ to mean that d is strongly valid.

– $\mathrm{SV}(\wedge^+(d,e))$ if $\mathrm{SV}(d)$ and $\mathrm{SV}(e)$.

– $\mathrm{SV}(\vee_i^+(d))$ if $\mathrm{SV}(d)$.

– $\mathrm{SV}(\exists^+(t,d))$ if $\mathrm{SV}(d)$.

– $\mathrm{SV}(\supset^+(u,d))$ if for any e with $\mathrm{SV}(e)$ we have $\mathrm{SV}(d[e/u])$.

– $\mathrm{SV}(\forall^+(x,d))$ if for any t we have $\mathrm{SV}(d[t/x])$.

– If d is not an introduction, then $\mathrm{SV}(d)$ if the following three conditions all hold.

(a) For all d' with $d \to d'$ we have $\mathrm{SV}(d')$.

(b) If $d = \vee^-(d_1, u_0, e_0, u_1, e_1)$, then for $i = 0$ and $i = 1$ we have $\mathrm{SV}(e_i)$ and for all d' with $d_1 \to^* d'$ and all e with $\mathrm{ES}(d', \vee_i^+(e))$ we have $\mathrm{SV}(e_i[e/u_i])$.

(c) If $d = \exists^-(d_1, x, u, e)$, then $\mathrm{SV}(e)$ and for all d' with $d_1 \to^* d'$ and all t, f with $\mathrm{ES}(d', \exists^+(t,f))$ we have $\mathrm{SV}(e[t,f/x,u])$.

Remark. In (c) a small oversight in Prawitz' formulation has been corrected by van de Pol. Prawitz (in [19], p. 293) does not mention that the top formula $\exists x\, B$ in the end segment of d' should have been obtained by $\exists^+$ from $B[t]$; this is needed in the proofs below. It could also have been obtained by

$$\frac{B[t] \to \exists x\, B \quad B[t]}{\exists x\, B}.$$

Lemma 4. *If $d \to d'$ and $\mathrm{SV}(d)$, then $\mathrm{SV}(d')$.*

Proof by induction on the definition of $\mathrm{SV}(d)$. If d is an introduction, then the claim follows immediately from the IH. If d is not an introduction, then by (a) in the definition of $\mathrm{SV}(d)$ we have $\mathrm{SV}(d')$. □

Lemma 5. *If $\mathrm{SV}(d)$, then $\mathrm{SN}(d)$.*

Proof by induction on the definition of $\mathrm{SV}(d)$. If d is an introduction, then the claim follows immediately from the IH and the fact that any assumption variable is strongly valid. If d is not an introduction, then by definition of $\mathrm{SV}(d)$ we have $\mathrm{SV}(e)$ for any e such that $d \to e$. By IH we have $\mathrm{SN}(e)$ for any such e, and hence $\mathrm{SN}(d)$. □

Lemma 6. *(i) If $\mathrm{ES}(d,e)$ and $\mathrm{SV}(d)$, then $\mathrm{SV}(e)$. (ii) If $\mathrm{ES}(d,e)$ and $e \to e'$, then there is a d' such that $\mathrm{ES}(d',e')$ and $d \to d'$. (iii) If $\mathrm{ES}(d,e)$ and $\mathrm{SN}(d)$, then $\mathrm{SN}(e)$.*

Proof. (i) is proved easily by induction over the definition of $\mathrm{ES}(d,e)$, using the first part of (b) and (c) in the definition of $\mathrm{SV}(d)$. (ii) is also proved easily by induction over the definition of $\mathrm{ES}(d,e)$. (iii) is an immediate consequence of (ii). □

Main Lemma 7. *If d is an elimination, then* $\mathrm{SV}(d)$ *if the following conditions hold.*

(i) $\mathrm{SN}(e)$ *for any immediate subderivation e of d.*

(ii) If the last rule of d is neither $\vee^-$ nor $\exists^-$, then $\mathrm{SV}(e)$ *for any immediate subderivation e of d.*

(iii) If the last rule of d is either $\vee^-$ or $\exists^-$, then conditions (b) and (c) of the definition of strong validity hold. More explicitely: (b) If $d = \vee^-(d_1, u_0, e_0, u_1, e_1)$, then for $i = 0$ and $i = 1$ we have $\mathrm{SV}(e_i)$ and for all d' with $d_1 \to^* d'$ and all e with $\mathrm{ES}(d', \vee_i^+(e))$ we have $\mathrm{SV}(e_i[e/u_i])$; (c) If $d = \exists^-(d_1, x, u, e)$, then $\mathrm{SV}(e)$ and for all d' with $d_1 \to^* d'$ and all t, f with $\mathrm{ES}(d', \exists^+(t, f))$ we have $\mathrm{SV}(e[t, f/x, u])$.

Proof. To any elimination d we assign an induction value, which is a triple (k, ℓ, m) of natural numbers where

$k =$ the length of the longest reduction sequence from the major premise of d,
$\ell =$ the depth of the major premise of d,
$m =$ the sum of the lengths of the longest reduction sequences from the immediate subderivations of d.

The induction values are ordered lexicographically.

Let d be an elimination such that (i)–(iii) hold. Let (k, ℓ, m) be the induction value. Note that k and m are finite by assumption (i). By the definition of $\mathrm{SV}(d)$ it suffices to show $\mathrm{SV}(d')$ for all d' such that $d \to d'$. We distinguish three cases.

I. d' is obtained from d by reducing a proper subderivation of d. Let (k', ℓ', m') be the induction value of d'. If the major premise of d was reduced, then $k > k'$. Otherwise $k = k'$, $\ell = \ell'$ and $m > m'$. So the induction value is lowered and we can use the IH, which says that it is enough to prove (i), (ii) and (iii) for d'. (i) follows from (i) for (d). (ii) follows from (ii) for (d), using Lemma 4. For (iii) we only treat the existential case. So let $d = \exists^-(d_1, x, u, e_1)$. Then $d' = \exists^-(d_2, x, u, e_2)$ with $d_1 \to d_2$ and $e_1 = e_2$, or $d_1 = d_2$ and $e_1 \to e_2$. We must show $\mathrm{SV}(e_2)$ and for all d' with $d_2 \to^* d'$ and all t, f with $\mathrm{ES}(d', \exists^+(t, f))$ we have $\mathrm{SV}(e_2[t, f/x, u])$. $\mathrm{SV}(e_2)$ holds, since by assumption d satisfies (iii) (we need Lemma 4 here in case $e_1 \to e_2$). Now let $d_2 \to^* d'$ and t, f with $\mathrm{ES}(d', \exists^+(t, f))$ be given. Then also $d_1 \to^* d'$. With (iii) for d it follows that $\mathrm{SV}(e_1[t, f/x, u])$. Then also $\mathrm{SV}(e_2[t, f/x, u])$ (we need Lemmata 1 and 4 here in case $e_1 \to e_2$).

II. d' is a proper reduct of d. Here the claim follows easily from the assumptions (i)–(iii) and the definition of strong validity.

Case $\supset^-(\supset^+(u, d), e) \to_0 d[e/u]$. We must prove $\mathrm{SV}(d[e/u])$. By (ii) we know $\mathrm{SV}(\supset^+(u, d))$ and $\mathrm{SV}(e)$. Hence $\mathrm{SV}(d[e/u])$ by the definition of strong validity.

Case $\wedge_i^-(\wedge^+(d_0, d_1)) \to_0 d_i$. We must prove $\mathrm{SV}(d_i)$. By (ii) we know $\mathrm{SV}(\wedge^+(d_0, d_1)))$. Hence $\mathrm{SV}(d_i)$ by the definition of strong validity.

Case $\forall^-(\forall^+(x,d),t) \to_0 d[t/x]$. We must prove $\mathrm{SV}(d[t/x])$. By (ii) we know that $\mathrm{SV}(\forall^+(x,d))$. Hence $\mathrm{SV}(d[t/x])$ by the definition of strong validity.

Case $\vee^-(\vee_i^+(d),u_0,e_0,u_1,e_1) \to_0 e_i[d/u_i]$. We must prove $\mathrm{SV}(e_i[d/u_i])$. This follows immediately from (iii), condition (b).

Case $\exists^-(\exists^+(t,d),x,u,e) \to_0 e[t,d/x,u]$. Similar.

III. d' is a permutative reduct of d. We only treat the case of an existential permutative conversion. Then

$$d = \Box^-(\exists^-(d_1,x,u,d_2),\vec{\alpha}),$$
$$d' = \exists^-(d_1,x,u,\Box^-(d_2,\vec{\alpha})).$$

Let (k',ℓ',m') be the induction value of d'. We will show in (i) below that k' and m' are finite. We have to prove $\mathrm{SV}(d')$. Note that the major premise of d' is an immediate subderivation of the major premise of d. Therefore $k \geq k'$ and $\ell > \ell'$. So the induction value is lowered and we can use the IH, which says that it is enough to prove (i) and (iii) for d'.

(i). $\mathrm{SN}(d_1)$ clearly follows from (i) for d. $\mathrm{SN}(\Box^-(d_2,\vec{\alpha}))$ follows by Lemma 5 from $\mathrm{SV}(\Box^-(d_2,\vec{\alpha}))$, which will be proved as $(*)$ below.

(iii). We have to show $(*)$ $\mathrm{SV}(\Box^-(d_2,\vec{\alpha}))$ and $(**)$ If $d_1 \to^* d_1'$ with $\mathrm{ES}(d_1',\exists^+(t,f))$, then $\mathrm{SV}(\Box^-(d_2[t,f/x,u],\vec{\alpha}))$. First note the following facts.

$$\mathrm{SN}(\vec{\alpha}), \quad \mathrm{SN}(\exists^-(d_1,x,u,d_2)) \quad \text{and} \quad \mathrm{SN}(d_2). \tag{F1}$$

This follows from assumption (i) for d. We also have

$$\text{If} \quad \Box \in \{\wedge,\supset,\forall\}, \quad \text{then} \quad \mathrm{SV}(\vec{\alpha}), \quad \mathrm{SV}(\exists^-(d_1,x,u,d_2)) \quad \text{and} \quad \mathrm{SV}(d_2). \tag{F2}$$

The first and second follow from assumption (ii) for d, and the third follows from the second by the definition of strong validity. Now we can prove $(*)$ and $(**)$.

$(*)$. Let (k_1,ℓ_1,m_1) be the induction value of $\Box^-(d_2,\vec{\alpha})$. Then $k \geq k_1$ and $\ell > \ell_1$. Hence we can apply the IH, which says that it suffices to prove (i)—(iii) for $\Box^-(d_2,\vec{\alpha})$. (i) follows from (F1) and (ii) follows from (F2). For (iii) we only treat the case $\Box^-(d_2,\vec{\alpha}) = \exists^-(d_2,y,v,d_3)$; the case with $\vee$ is similar. Then

$$d = \exists^-(\exists^-(d_1,x,u,d_2),y,v,d_3),$$
$$d' = \exists^-(d_1,x,u,\exists^-(d_2,y,v,d_3)).$$

We have to prove $\mathrm{SV}(d_3)$ and for all d_2' with $d_2 \to^* d_2'$ and all s,g such that $\mathrm{ES}(d_2',\exists^+(s,g))$ we have $\mathrm{SV}(d_3[s,g/y,v])$. The first follows from (c_1) for d. For the second assume $d_2 \to^* d_2'$ and $\mathrm{ES}(d_2',\exists^+(s,g))$. Then $\mathrm{SV}(d_3[s,g/y,v])$ follows from (c_2) for d, since $\exists^-(d_1,x,u,d_2) \to^* \exists^-(d_1,x,u,d_2')$ and $\mathrm{ES}(\exists^-(d_1,x,u,d_2'),\exists^+(s,g))$.

$(**)$. Let $d_1 \to^* d_1'$ and $\mathrm{ES}(d_1', \exists^+(t,f))$. We must show $\mathrm{SV}(\Box^-(d_2[t,f/x,u],\vec{\alpha}))$. Let (k_2, ℓ_2, m_2) be the induction value of $\Box^-(d_2[t,f/x,u],\vec{\alpha})$. We will need yet another fact:

$$\text{There is an } e \text{ such that} \quad \exists^-(d_1,x,u,d_2) \to^+ e \quad \text{and} \quad \mathrm{ES}(e, d_2[t,f/x,u]). \qquad (F3)$$

This follows from Lemma 2, since $d_1 \to^* d_1'$ and $\mathrm{ES}(d_1', \exists^+(t,f))$. From (F3) we immediately obtain $k > k_2$, using Lemma 6. Clearly m_2 is finite by (F1). So the induction value is lowered and we can use the IH, which says that it is enough to prove (i), (ii) and (iii) for $\Box^-(d_2[t,f/x,u],\vec{\alpha})$.

(i). $\mathrm{SN}(\vec{\alpha})$ holds by (F1). $\mathrm{SN}(d_2[t,f/x,u])$ follows from (F3), Lemma 6(iii) and (F1).

(ii). Let $\Box \in \{\wedge, \supset, \forall\}$. Then we obtain $\mathrm{SV}(\vec{\alpha})$ by (F2). $\mathrm{SV}(d_2[t,f/x,u])$ follows from $\mathrm{SV}(\exists^-(d_1,x,u,d_2))$ (which holds by (F2)) by (c) in the definition of strong validity (since $d_1 \to^* d_1'$ and $\mathrm{ES}(d_1', \exists^+(t,f))$).

(iii). We only treat the case $\Box^-(d_2,\vec{\alpha}) = \exists^-(d_2,y,v,d_3)$; the case with $\vee$ is similar. Then

$$d = \exists^-(\exists^-(d_1,x,u,d_2),y,v,d_3),$$
$$d' = \exists^-(d_1,x,u,\exists^-(d_2,y,v,d_3)).$$

We have to prove $\mathrm{SV}(d_3)$ and for all d_2' with $d_2[t,f/x,u] \to^* d_2'$ and all $s.g$ such that $\mathrm{ES}(d_2', \exists^+(s,g))$ we have $\mathrm{SV}(d_3[s,g/y,v])$. The first follows from (c_1) for d. For the second assume $d_2[t,f/x,u] \to^* d_2'$ and $\mathrm{ES}(d_2', \exists^+(s,g))$. We must show $\mathrm{SV}(d_3[s,g/y,v])$. By (F3) we have an e such that

$$\exists^-(d_2,x,u,d_1) \to^+ e \quad \text{and} \quad \mathrm{ES}(e, d_2[t,f/x,u]).$$

But then, because of $d_2[t,f/x,u] \to^* d_2'$, Lemma 6(ii) gives us an e' such that

$$\exists^-(d_2,x,u,d_1) \to^+ e' \quad \text{and} \quad \mathrm{ES}(e', d_2').$$

Because of $\mathrm{ES}(d_2', \exists^+(s,g))$ and the transitivity of the end segment relation we have $\mathrm{ES}(e', \exists^+(s,g))$. Therefore the main premise of d reduces to e' with $\exists^+(s,g)$ in its end segment. By (c_2) for d we can conclude that the side premise of d with s,g substituted is strongly valid, i.e. $\mathrm{SV}(d_3[s,g/y,v])$. $\Box$

A derivation d is called *strongly valid under substitution* (written $\mathrm{SV}^*(d)$) if for any object terms $\vec{t}$ and strongly valid derivations $\vec{f}$ we have $\mathrm{SV}(d[\vec{t},\vec{f}])$. Using the Main Lemma 7 we can now prove

Lemma 8. *Any derivation d is strongly valid under substitution.*

Proof by induction over d. *Case u.* Then $\mathrm{SV}(u)$ by definition.

Cases $\wedge^+(d,e)$, $\vee_i^+(d)$, $\exists^+(t,d)$. The claim follows immediately from the definition of strong validity and the IH.

Case $\supset^+(u, d)$. Let $\vec{f}$ be strongly valid. We have to show that $\supset^+(u, d[\vec{t}, \vec{f}])$ is strongly valid. So let e be strongly valid. We have to show that $d[e, \vec{f}]$ is strongly valid. But this follows from the IH for d.

Case $\forall^+(x, d)$. Similar.

Case $\Box^-(\vec{\alpha})$. Let $\vec{f}$ be strongly valid. We have to prove (i), (ii) and (iii) of the Main Lemma 7 for $\Box^-(\vec{\alpha})[\vec{t}, \vec{f}]$. By IH $\mathrm{SV}(\alpha_i[\vec{t}, \vec{f}])$ for every immediate subderivation α_i, and by Lemma 5 we can conclude that $\mathrm{SN}(\alpha_i[\vec{t}, \vec{f}])$. This proves (i) and (ii). For (iii) we have to prove conditions (a) and (b) of the definition of strong validity. We only treat (c). So let $d = \exists^-(d_1, x, u, e)$, hence $d[\vec{t}, \vec{f}] = \exists^-(d_1[\vec{t}, \vec{f}], x, u, e[\vec{t}, \vec{f}])$. Let $d_1[\vec{t}, \vec{f}] \to^* d'$ and $\mathrm{ES}(d', \exists^+(t, f))$. We must show $\mathrm{SV}(e[t, \vec{t}, \vec{f}])$. Since $\mathrm{SV}(d_1[\vec{t}, \vec{f}])$ by IH, we get $\mathrm{SV}(d')$ by Lemma 4, hence $\mathrm{SV}(\exists^+(t, f))$ by Lemma 6(i) and hence $\mathrm{SV}(f)$ by the definition of strong validity. Now from the IH for e we obtain $\mathrm{SV}(e[t, \vec{t}, f, \vec{f}])$. $\Box$

From Lemma 5 and Lemma 8 we immediately obtain

Theorem. $\to$ *is terminating, i.e. any derivation d is strongly normalizable.* $\Box$

References

[1] Heinz Bachmann. *Transfinite Zahlen.* Springer, Berlin, 1955.

[2] Ulrich Berger and Helmut Schwichtenberg. An inverse of the evaluation functional for typed λ–calculus. In Rao Vemuri, editor, *Proceedings of the Sixth Annual IEEE Symposium on Logic in Computer Science*, pages 203–211. IEEE Computer Society Press, Loss Alamitos, 1991.

[3] Kim B. Bruce, Albert R. Meyer, and John C. Mitchell. The semantics of second order lambda-calculus. *Information and Computation*, 85:76–134, 1990.

[4] William Clinger, Jonathan Rees (editors) H. Abelson, N.I. Adams IV, D.H. Bartley, G. Brooks, R.K. Dybvig, D.P. Friedman, R. Halstead, C. Hanson, C.T. Haynes, E. Kohlbecker, D. Oxley, K.M. Pitman, G.J. Rozas, G.L. Steele Jr., G.J. Sussman, and M. Wand. *Revised^4 Report on the Algorithmic Language Scheme*, 1991. Appeared in ACM Lisp Pointers IV, July-September 1991, and also as MIT AI Memo 848b. It can be obtained by anonymous ftp at the two Scheme Repositories, altdorf.ai.mit.edu and nexus.yorku.ca.

[5] Catarina Coquand. From semantics to rules: A machine assisted analysis. Submitted to CSL'93, September 1993.

[6] Roberto Di Cosmo and Delia Kesner. A confluent reduction for the extensional typed λ–calculus with pairs, sums, recursion and terminal object. In *ICALP 93*, number 700 in Lecture Notes in Computer Science, pages 645–656, Heidelberg, 1993. Springer.

[7] Roberto Di Cosmo and Delia Kesner. Simulating expansions without expansions. Rapport de recherche 1911, INRIA, Roquencourst, May 1993.

[8] Daniel J. Dougherty. Some lambda calculi with categorical sums and products. In *Proceedings of the Fifth International Conference on Rewriting Techniques and Applications (RTA)*, number 700 in Lecture Notes in Computer Science, pages 137–151. Springer, 1993.

[9] Yuri L. Ershov. Model C of partial continuous functionals. In R. Gandy and M. Hyland, editors, *Logic Colloquium 1976*, pages 455–467. North Holland, Amsterdam, 1977.

[10] Harvey Friedman. Classically and intuitionistically provably recursive functions. In Dana S. Scott and Gert H. Müller, editors, *Higher Set Theory*, pages 21–28. Springer Lecture Notes in Mathematics, Volume 699, 1978.

[11] Gerhard Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1934.

[12] Gerhard Gentzen. Beweisbarkeit und Unbeweisbarkeit von Anfangsfällen der transfiniten Induktion in der reinen Zahlentheorie. *Mathematische Annalen*, 119:140–161, 1943.

[13] G.E. Mints. Exact estimates of the provability of transfinite induction in the initial segments of arithmetic. *Journal of Soviet Math*, 1:85–91, 1973. Translated from Zapiski Nauch. Sem. Leningrad *20*, 134–144 (1971).

[14] G.E. Mints. On e–theorems (in russian). *Zapiski*, 40:110–118, 1974.

[15] M.H.A. Newman. On theories with a combinatorial definition of "equivalence". *Annals of Mathematics*, 43(2):223–243, 1942.

[16] V. P. Orevkov. Lower bounds for increasing complexity of derivations after cut elimination. *Zapiski Nauchnykh Seminarov Leningradskogo*, 88:137–161, 1979.

[17] Charles Parsons. Transfinite induction in subsystems of number theory (abstract). *Journal of Symbolic Logic*, 38(3):544–545, 1973.

[18] Gordon D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–255, 1977.

[19] Dag Prawitz. Ideas and results in proof theory. In Jens Erik Fenstad, editor, *Proceedings of the second Scandinavian Logic Symposium*, pages 235–307. North–Holland, Amsterdam, 1971.

[20] Helmut Schwichtenberg. Normalization. In F.L. Bauer, editor, *Logic, Algebra and Computation. Proceedings of the International Summer School Marktoberdorf, Germany, July 25 – August 6, 1989*, Series F: Computer and Systems Sciences, Vol. 79, pages 201–237, Berlin, 1991. NATO Advanced Study Institute, Springer.

[21] Helmut Schwichtenberg. Primitive recursion on the partial continuous functionals. In Manfred Broy, editor, *Informatik und Mathematik*, pages 251–269. Springer, Berlin, 1991.

[22] Helmut Schwichtenberg. Minimal from classical proofs. In E. Börger, G. Jäger, H. Kleine-Büning, and M.M. Richter, editors, *Computer Science Logic*, pages 326–328. Springer LNCS 626, 1992.

[23] Helmut Schwichtenberg. Proofs as programs. In Peter Aczel, Harold Simmons, and Stanley S. Wainer, editors, *Proof Theory. A selection of papers from the Leeds Proof Theory Programme 1990*, pages 81–113. Cambridge University Press, 1992.

[24] Helmut Schwichtenberg. Density and choice for partial continuous functionals. In preparation, 1993.

[25] Helmut Schwichtenberg. Logikprogrammierung. Vorlesungsmanuskript, Universität München, 1994.

[26] Helmut Schwichtenberg and Stan Wainer. Ordinal bounds for programs. Submitted for publication in: Feasible Mathematics II (ed. Jeff Remmel), August 1993.

[27] Dana Scott. Domains for denotational semantics. In E. Nielsen and E. M. Schmidt, editors, *Automata, Languages and Programming*, Lecture Notes in Computer Science, Volume 140, pages 577–613. Springer, Berlin, 1982. A corrected and expanded version of a paper prepared for ICALP'82, Aarhus, Denmark.

[28] Martin Stein. *Interpretationen der Heyting–Arithmetik endlicher Typen.* PhD thesis, Universität Münster, Fachbereich Mathematik, 1976.

[29] William W. Tait. Infinitely long terms of transfinite type. In J. Crossley and M. Dummett, editors, *Formal Systems and Recursive Functions*, pages 176–185, Amsterdam, 1965. North–Holland.

[30] Anne S. Troelstra, editor. *Metamathematical Investigations of Intuitionistic Arithmetic and Analysis*, volume 344 of *Lecture Notes in Mathematics.* Springer, Berlin, 1973.

[31] Anne S. Troelstra and Dirk van Dalen. *Constructivism in Mathematics. An Introduction*, volume 121, 123 of *Studies in Logic and the Foundations of Mathematics.* North–Holland, Amsterdam, 1988.

[32] Jaco van de Pol. Stong normalization of fol with permutative conversions. Manuscript, May 1994.

[33] Anton Wallner. Komplexe Existenzbeweise in der Arithmetik. Master's thesis, Mathematisches Institut der Universität München, 1993.

[34] Hermann Weyl. Über die neue Grundlagenkrise der Mathematik. *Mathematische Zeitschrift*, 10, 1921.

Index

$\exists^*$–rules, 39
$\vee^*$–rules, 41

almost normal, 84
assumption variable, 6
atom, 2
atomic formula, 2

beta–eta–conversion, 15
beta–eta–normal, 16
branch, 20

classical logic, 9
closed, 2
closed derivation term, 8
Coincidence Lemma, 5
constant, 1
conversion relation, 15
critical, 65

decidable, 51, 65
deduction theorem, 13
definite Horn formula, 25
delta–expansion, 23
derivable formula, 8
derivation term, 6
disjunction property, 41

end segment, 88

environment, 4
eta–expansion, 21
eta–expansor, 22
Ex–falso–quodlibet axiom, 10
Ex–falso–quodlibet Lemma, 11
existence property, 41
Existence–Elimination–Lemma, 35
Existence–Introduction–Lemma, 35
extensionality axioms, 52

formula, 1
free assumption variable, 6
function symbol, 1

generalized definite Horn formula, 25

Harrop formula, 43
Herbrand's theorem, 20
Heyting, 38
Hilbert system, 13
Horn formula, 25

immediate reduct, 87
immediate subformula, 20
indirect proof, 10
inner delta–expansion, 23
instance, 41
intuitionistic logic, 9

judgement, 43

level, 1, 76
long normal form, 21

minimal logic, 6
Mints formula, 10
modified realizability interpretation, 43

natural deduction, 6
normalization by evaluation, 32
number, 49

one–step reduction relation, 16, 87
Orevkov, 34

parallel reduction relation, 27
Peano–axioms, 76
Peirce formula, 10
permutative reduct, 87
position, 19
  elimination, 19
  end, 19
  independent, 19
  introduction, 19
  leaf, 19
  minimal, 19
progression rule, 83
progressive, 78
propositional symbol, 2

relation symbol, 2
relevant, 66

SLD–Resolution, 25
stability axiom, 10
Stability Lemma, 10

strong computability predicates, 16
strong delta–expansion, 23
strong disjunction, 40
strong existential quantifier, 38
strongly valid, 88
strongly valid under substitution, 92
subformula, 20
subformula property, 20
substitution, 3
Substitution Lemma, 5
surjectivity of pairing, 27

term, 1
type, 1
types associated with A, 43

uniqueness of normal forms, 27

variable, 1

Weyl, 38