Programmextraktion für nicht-wachsende Polynomialzeitberechnungen

Klaus Aehlig

2. Mai 2000

Zusammenfassung

Es wird ein rein syntaktischer Beweis gegeben, daß alle in einer affin linear getypten Variante von Gödels T definierbaren Funktionen polynomialzeitberechenbar sind. Dieses Termsystem erlaubt Iteration in allen endlichen Typen. Der Beweis gibt explizite, durch Induktion über den Termaufbau definierte polynomiale Schranken an.

Ferner wird eine Variante der Heytingschen Arithmetik vorgeschlagen, bei der alle realisierenden Terme in dem vorgestelleten Termsystem liegen und insbesondere also polynomialzeitberechenbar sind. Die Realisierbarkeitsaussage ist in dem betrachteten Arithmetiksystem, also insbesondere in der üblichen Heytingschen Arithmetik beweisbar.

Ich erkläre hiermit, daß ich diese Arbeit selbständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Taufkirchen, den 2. Mai 2000

(Klaus Aehlig)

Einführung

Von allen betrachteten Funktionenklassen ist `Ptime`, also die Menge aller in polynomialer Zeit berechenbaren Funktionen von besonderem Interesse, da diese Funktionen nicht nur prinzipiell, sondern auch mit akzeptablem Aufwand berechnet werden können. Es ist daher oft von Interesse, von sich natürlich ergebenden Funktionen zu wissen, daß sie in `Ptime` liegen. Eine Möglichkeit dies sicherzustellen ist es, die Definierbarkeit in einem Termsystem nachzuweisen, in dem nur `Ptime`–Funktionen definierbar sind. Das Studium solcher Termsysteme ist auch für sich genommen von Interesse, da auf diese Weise ein besseres Verständnis der Ursachen der Komplexität von Algorithmen gewonnen werden kann; dies ist insbesondere dann der Fall, wenn die polynomialen Schranken explizit gegeben sind. Im ersten Kapitel dieser Diplomarbeit wird ein solches, auf Hofmann [11] zurückgehendes Termsystem vorgestellt und ein syntaktischer Beweis gegeben, daß alle definierbaren Funktionen in `Ptime` liegen. Auf diese Weise werden auch explizite Schranken gewonnen; dieses Kapitel ist ferner als "ausführliche Fassung" von [1] zu sehen.

Ein in der Logik besonders wichtiges Beispiel für sich ergebende berechenbare Funktionen ist die Programmextraktion: Beweist man in konstruktiver Arithmetik eine Formel der Form $\forall x \exists^* y A$ (etwa mit A quantorenfrei), so kann man daraus ein "Programm" (gegeben durch einen Funktionsterm) extrahieren, das diese Formel "realisiert", das also zu gegebenem x ein passendes y findet. Diese Programme sind von ihrer Entstehung her beweisbar korrekt (und der Korrektheitsbeweis kann sogar im ursprünglichen Arithmetiksysem geführt werden!), die Programme liegen jedoch im allgemeinen in Gödels T, einem Termsystem, in dem auch Funktionen mit großer Komplexität definierbar sind. Um von *praktischem* Wert zu sein, ist es bei solchen extrahierten Termen daher oft notwendig, Laufzeitschranken garantieren zu können. Dazu muß die Arithmetik eingeschränkt werden, da in dieser etwa die Existenz der Funktion $x \mapsto 2^x$, also einer Funktion mit exponentiellem Wachstum beweisbar ist. Ein Vorschlag, wie eine solche Einschränkung aussehen könnte, wird im zweiten Kapitel vorgestellt. Alle realisierenden Terme liegen im Hofmannschen Termsystem und damit insbesondere in `Ptime`. Die Realisierbarkeitsaussage ist sogar in dem eingeschränkten System beweisbar (und zu einem gegebenem Beweis kann das extrahierte Programm in polynomialer Zeit gefunden werden).

Im letzten Kapitel werden abschließend noch mögliche Erweiterungen des zu Grunde liegenden Termsystems vorgestellt. Auch wenn dies nicht explizit diskutiert wird, so sind diese Erweiterungen stets vor dem Hintergrund zu lesen, daß diese Erweiterungen die Grundlage für ein Arithmetiksystem bilden, das die entsprechenden wünschenswerten Eigenschaften aufweist, aber trotzdem noch `Ptime`–Realisierungen besitzt.

Inhaltsverzeichnis

1 Ein syntaktischer Beweis für die polynomiale Normalisierbarkeit von Hofmanns System $\mathcal{C}$

In letzter Zeit sind viele Charakterisierungen von `Ptime`, also der Klasse aller in polynomialer Zeit berechenbaren Funktionen[1] gefunden worden, die durch Einschränkung aus Gödels T [8] entstehen [2, 3, 10].

Betrachten wir das folgende Beispiel für unär codierte natürliche Zahlen[2], welches zu exponentiellem Wachstum führt.

$$
\begin{aligned}
\mathtt{double}(0) &= 0 \\
\mathtt{double}(Sx) &= SS\mathtt{double}(x)
\end{aligned}
$$

$$
\begin{aligned}
\mathtt{exp}(0) &= S0 \\
\mathtt{exp}(Sx) &= \mathtt{double}(\mathtt{exp}(x))
\end{aligned}
$$

Auf "predicative recursion" [2, 13] basierende Ansätze sehen den Grund für dieses Wachstum (und damit die lange Laufzeit) in der Art, in der `double` aufgerufen wird: der vorhergehende Wert einer äußeren Rekursion, nämlich $\mathtt{exp}(x)$ wird als rekursives Argument einer inneren Rekursion, nämlich `double` verwendet.

Auch wenn in solchen Systemen alle Polynomialzeitbrechenbaren *Funktionen* definierbar sind, so scheiden doch einige natürliche *Algorithmen* aus. "Insertion sort" hat etwa die gleiche rekursive Struktur wie `exp`:

$$
\begin{aligned}
\mathtt{insert}(a, []) &= [a] \\
\mathtt{insert}(a, [b \mid \ell]) &= [b' \mid \mathtt{insert}(a', \ell)] \quad a', b' \text{ eine Permutation von } a, b \text{ mit } b' \leq a'
\end{aligned}
$$

$$
\begin{aligned}
\mathtt{sort}([]) &= [] \\
\mathtt{sort}([a \mid \ell]) &= \mathtt{insert}(a, \mathtt{sort}(\ell))
\end{aligned}
$$

Caseiro [7] beobachtete dies und studierte zahlreiche verwandte Beispiele. Ihre Arbeit führte zu einigen (zum Teil semantischen) Kriterien, die polynomiale Laufzeit sicherstellen. Im Fall von "insertion sort" liegt der Grund, daß diese Rekursion nicht zu exponentiellem Wachstum führt daran, daß `insert` seine Eingabe nicht vergrößert. Hofmann [11] führte diese Idee fort und formulierte ein Termsystem (im folgendem

[1] `Ptime` bezeichnet die Menge aller Funktionen für die es einen Algorithmus (gegeben etwa durch die Kodierung einer Turingmaschine) und ein Polynom gibt, so daß der Algorithmus auf beliebige Eingaben stets mit der richtigen Antwort terminiert und die Zahl der benötigten Rechenschritte nach oben durch das Polynom, ausgewertet an der Stelle "Zahl der Zeichen der Eingabe" beschränkt ist. Da für das Verständnis dieser Arbeit, mit Ausnahme des Kapitels 3.1 keine exakte Definition von `Ptime` nötig ist, sei für die genaue Definition auf die Literatur (etwa [15, §16,§23]) verwiesen.

[2] Natürlich verwendet niemand bei tatsächlichen Komplexitätsbetrachtungen unär codierte natürliche Zahlen. Trotzdem sind Beispiele mit unär codierten natürlichen Zahlen von sehr allgemeiner Natur, da die meisten natürlich vorkommenden Datenstrukturen durch geeignetes "Vergessen von Details" als unär codierte natürliche Zahlen aufgefaßt werden können: so sind etwa binär codierte natürliche Zahlen nach Vergessen des Unterschieds der beiden Nachfolgerfunktionen unär codiert; ferner können Listen als unär codierte natürliche Zahlen aufgefaßt werden, wenn die Elemente vergessen werden.

dem Vorschlag von [5] folgend zur Ehre von Caseiro mit $\mathcal{C}$ bezeichnet) mit einem speziellen Typ $\diamond$ von "Münzen" in dem nur nicht-wachsende `Ptime` Funktionen definierbar sind, das aber verschachtelte Iterationen zuläßt. Die Grundidee ist, daß für die Verwendung einer Funktion die die Größe ihres Arguments vergrößert, wie etwa die binären Nachfolgerfunktionen, eine Münze (in Form eines Terms vom Typ $\diamond$) bezahlt werden muß.

Der Beweis, daß alle definierbaren Funktionen die gewünschten Eigenschaften haben, verwendet dabei die mengentheoretische Interpretation von Termen, also ein semantisches Konzept. In diesem Kapitel wird ein rein syntaktischer Beweis dieses Resultats angegeben. Dazu wird zunächst (eine leichte Modifikation von) Hofmanns Termsystem vorgestellt, auf diesem werden (in Abschnitt 1.2) Reduktionsregeln eingeführt und anschließend wird bewiesen, daß die Zahl der Reduktionsschritte zur Berechnung der Normalform polynomial beschränkt ist. Die beschränkenden Polynome werden dabei (in Definition 1.3.9) explizit definiert.

Auf diese Weise wird auch eine schöne Deutung der Hofmannschen "Münzen" gefunden: Diese entsprechen gerade den freien Variablen. Lemma 1.3.3 wird zeigen, daß jeder Term vom Typ $\diamond$ eine freie Variable enthält. Das "bezahlen einer Münze" impliziert also den Einsatz einer freien Variablen — und diese sind im Kontext (affin) linearer Logik ja eine knappe Resource. Bei dieser Deutung der Zahl der freien Variablen als "richtigen semantischen Längenbegriff" (anstelle der "size function" aus [11]) wird auch die Eigenschaft des Nicht-wachsens trivial: Keine der Reduktionsregeln aus Definition 1.2.2 führt neue Variablen ein.

1.1 Das Termsystem $\mathcal{C}$

Ausgehend von den Basistypen $\mathbf{N}$ (natürliche Zahlen) und bool (boolesche Werte) sowie dem "Längen"-typ $\diamond$ werden induktiv die Typen endlicher Ordnung gebildet. Das besondere am Typ $\diamond$ ist, daß es *keine* Konstruktorsymbole für diesen Typ gibt. Lemma 1.3.3 wird zeigen, daß deshalb zur Bildung eines Terms dieses Typs eine freie Variable erforderlich ist. Da freie Variable im verwendeten Kontext linearer Logik ein geeignetes Längenmaß bilden, können durch die Verwendung dieses Typs also Bedingungen an die Länge von Termen beziehungsweise das "Vorhandensein von Arbeitsspeicher"[3] formuliert werden.

Definition 1.1.1 (lineare endliche Typen). Die linearen endlichen Typen sind induktiv definiert[4]:

$\tau, \rho ::= \ \diamond \mid \mathbf{N} \mid \mathrm{bool} \mid \tau \multimap \rho \mid \tau \otimes \rho \mid \tau \times \rho \mid \mathbf{L}(\tau)$

Dabei bezeichnet $\mathbf{L}(\tau)$ den Typ der Listen über τ. $\tau \otimes \rho$ und $\tau \times \rho$ können beide als geeordnete Paare interpretiert werden. Der Unterschied liegt jedoch in der Bedeutung

[3] bei einer Übersetzung in eine procedurale Programmiersprache; vergleiche hierzu [9]

[4] Diese in der Informatik übliche und im folgenden stets stillschweigend verwandte Aufschreibung einer strikt positiven Definition ist wie folgt zu lesen: "$\diamond$, $\mathbf{N}$ und bool sind Typen. Sind τ und ρ schon Typen, so auch $\tau \multimap \rho$, $varTypeA \otimes \rho$, $\tau \times \rho$ und $\mathbf{L}(\tau)$. Außerdem ist die Menge aller Typen die kleinste Menge mit diesen Eigenschaften."

von “linear” für diese Typen: lineare Verwendung eines Tensortyps (also von $\tau \otimes \rho$) bedeutet einmalige Verwendung jeder Komponente, während lineare Verwendung eines Produkts von Typ $\tau \times \rho$ es einem lediglich erlaubt, das Paar als solches einmal zu verwenden. Insbesondere kann von einem gewöhnlichen Paar lediglich eine der beiden Komponenten betrachtet werden. Wie aus Definition 1.2.2 ersichtlich korrespondieren $\otimes$ und $\multimap$, sowie bool und $\times$. Als Mitteilungszeichen für Typen werden $\tau, \rho, \sigma, \ldots$ verwendet.

Ausgangspunkt für die Bildung von Termen sind Variablen und Konstruktorsymbole. Variablen werden durch $x, y, z, l, p, q, \ldots$ mitgeteilt. Jede Variable hat einen ihr zugehörigen Typ. Die Schreibweise x^τ bezeichne die Variable x; sie kann jedoch nur gebildet werden, wenn die Variable x den Typ τ hat. Für jeden Typ (Definition 1.1.1) seien unendlich viele Variablen dieses Typs vorhanden. Die Menge der Konstruktorsymbole wird in Definition 1.1.3 festgelegt. Mitteilungszeichen für Konstruktorsymbole sind $c, \ldots$. Im folgenden seien $a, h, t, s, r, \ldots$ Mitteilungszeichen für Terme.

Definition 1.1.2 (Terme). Terme werden induktiv definiert durch:

$t, s ::= \ x \mid c \mid \lambda x^\tau t \mid \langle t, s\rangle \mid ts \mid \{t\} \mid \{t, s\}$

Dabei stehen $\lambda x^\tau t$ und $\langle t, s\rangle$ für Funktionsdefinition durch λ-Abstraktion und Definition von Paaren durch Angabe der Komponenten. ts wird allgemein für Eliminationen (also nicht nur $\multimap$-Eliminationen!) verwendet. So wird etwa die Iteration über eine Liste ℓ als $\mathbf{L}(\tau)$-Elimination verstanden und als $\ell\,\{s\}\,t$ geschrieben. Dabei bezeichntet s den Schrittfall und t den Anfangsfall. Die Idee, Eliminationen einheitlich als Applikationen zu schreiben, ist [12] entnommen. Die durch $\{\cdot\}$ und $\{\cdot,\cdot\}$ gebildeten Terme sind in dieser Tradition zu sehen. In der hier verwendeten Form markieren sie die Schrittterme bei Iteration über binäre Numerale bzw. Listen. Insbesondere können derartige Terme nur als rechte Komponente einer Applikation auftreten; vergleiche hierzu auch Lemma 1.1.8. Diese explizite[5] Kennzeichnung der Schrittterme erlaubt es, gewisse Definitionen über den Aufbau von Termen, statt über den Beweis der Typisiertheit zu führen. Dies ist insofern flexibler, als auch durch $\{\cdot\}$ gebildeten Termen Werte zugewiesen werden können, obwohl diese nicht typisiert sind. Dadurch werden etwa die den Termen zugewiesenen polynomialen Schranken begrifflich klarer: Ein Zeichen außerhalb von $\{\cdot\}$ trägt zum konstanten Glied des die Laufzeit bestimmenden Polynoms bei, ein Zeichen innerhalb einer $\{\cdot\}$ zum Koeffizienten von X, ein Zeichen innerhalb von zwei geschachtelten $\{\cdot\}$ zum Koeffizienten von X^2, und so weiter. Beachte hierzu die Definition 1.3.9 von $\vartheta(\cdot)$.

Endliche Listen $t_1, \ldots, t_n$ mit $n \geq 0$ werden auch durch Vektoren $\vec{t}$ mitgeteilt. Es bezeichne $\|\vec{t}\|$ die Zahl der Elemente des Vektors $\vec{t}$; es ist also $\|t_1, \ldots, t_n\| = n$. Von der üblicheren Schreibweise $|\vec{t}|$ wird wegen des Notationskonflikts mit der in Definition 1.3.1 eingeführten Schreibweise für die Zahl der Zeichen abgewichen. Die Schreibweisen $t\vec{t}$ und $\lambda\vec{x}^{\vec{\tau}}\,t$ sind in der naheliegenden Weise definiert: da Applikationen stets linksgeklammert zu lesen sind, sind darunter also $(((tt_1)t_2)\ldots t_n)$ und

[5] “Explizit” in dem Sinne, daß die $\{\cdot\}$ in den “offiziellen” Termen auftauchen und nicht nur gedachte Zusatzmarkierungen seine mögen.

$\lambda x_1^{\tau_1}(\lambda x_2^{\tau_2}(\ldots(\lambda x_n^{\tau_n} t)))$ zu verstehen. Insbesondere ist $\lambda x^\tau, y^\rho\, t$ eine Abkürzung für $\lambda x^\tau(\lambda y^\rho t)$.

Außerdem werden Terme identifiziert, die sich nur durch Umbenennung der gebundenen Variablen (durch Variablen desselben Typs) unterscheiden. Dabei mögen in $\lambda x^\tau t$ alle freien Vorkommen von x in t durch λx^τ gebunden werden. Dabei ist der Geltungsbereich einer λ-Abstraktion als "so weit wie möglich" zu lesen. Es ist also etwa $\lambda x\, tx$ kurz für $\lambda x(tx)$.

Definition 1.1.3 (Konstruktorsymbole). Die Konstruktorsymbole und ihre Typen sind:

$\mathbf{t}$	bool
$\mathbf{f}$	bool
0	$\mathbf{N}$
$\mathbf{S}_0$	$\diamond \multimap \mathbf{N} \multimap \mathbf{N}$
$\mathbf{S}_1$	$\diamond \multimap \mathbf{N} \multimap \mathbf{N}$
$\mathbf{nil}_\tau$	$\mathbf{L}(\tau)$
$\mathbf{cons}_\tau$	$\diamond \multimap \tau \multimap \mathbf{L}(\tau) \multimap \mathbf{L}(\tau)$
$\otimes_{\tau,\rho}$	$\tau \multimap \rho \multimap \tau \otimes \rho$

Dabei ist die intendierte Bedeutung von 0 die Zahl 1 und die $\mathbf{S}_i$ stellen die binären Nachfolger (also $x \mapsto 2 \cdot x + i$) da (damit sind die natürlichen Zahlen also eindeutig als binäre Wörter codiert[6]).

Bei auf "type assignment" basierenden Ansätzen beschreiben Kontexte die Typisierung der in einem Term freien Variablen. Da es im vorliegenden Ansatz genügt, sich auf getypte Variablen zu beschränken, vereinfacht sich der Begriff Kontext zu folgender Definition:

Definition 1.1.4 (Kontext). Ein *Kontext* ist eine Menge von Variablen.

Für Kontexte werden die Mitteilungszeichen $\Gamma, \Gamma_1, \Gamma_2, \ldots$ verwendet. Die Schreibweise Γ_1, Γ_2 bezeichne $\Gamma_1 \cup \Gamma_2$, kann aber nur gebildet werden, wenn Γ_1 und Γ_2 disjunkt sind. Für einelementige Kontexte $\{x^\tau\}$ wird auch x^τ geschrieben.

Die folgende Definition gibt wieder, welche Terme richtig (das heißt affin linear) getypt sind. Dabei ist $\Gamma \vdash t^\tau$ zu lesen als "t ist ein getypter Term, sein Typ ist τ und seine freien Variablen sind in Γ enthalten". Die Notation für die Eliminationsregeln ist an [12] angelehnt. Bei den Regeln $(\mathbf{N}^-)$ und $(\mathbf{L}(\tau)^-)$ wird verlangt, daß die Schritterme geschlossen sind. Diese Forderung ist nötig, da die Schritterme beim "Ausfalten" der Iteration ja verdoppelt werden und das System andernfalls nicht unter Reduktion abgeschlossen wäre.

[6] Legt man darauf Wert, daß die natürlichen Zahlen mit 0 beginnen, so stelle man sich unter dem Zeichen 0 die natürliche Zahl 0 vor und unter $\mathbf{S}_i$ die Abbildung $x \mapsto 2 \cdot x + 1 + i$. Damit erhält man $\mathbf{S}_0 0$ als Kode der 1, $\mathbf{S}_1 0$ als Kode der 2, $\mathbf{S}_0(\mathbf{S}_0 0)$ als Kode der 3, und so weiter.

Definition 1.1.5 ($\Gamma \vdash t^\tau$). Die Relation $\Gamma \vdash t^\tau$ wird induktiv definiert[7].

$$\frac{}{\Gamma, x^\tau \vdash x^\tau} \qquad (Var)$$

$$\frac{c \text{ vom Typ } \tau}{\Gamma \vdash c^\tau} \qquad (const)$$

$$\frac{\Gamma \cup \{x^\tau\} \vdash t^\rho}{\Gamma \vdash (\lambda x^\tau t)^{\tau \multimap \rho}} \qquad (\multimap^+)$$

$$\frac{\Gamma \vdash t^\tau \qquad \Gamma \vdash s^\rho}{\Gamma \vdash \langle t, s\rangle^{\tau \times \rho}} \qquad (\times^+)$$

$$\frac{\Gamma_1 \vdash t^{\tau \multimap \rho} \qquad \Gamma_2 \vdash s^\tau}{\Gamma_1, \Gamma_2 \vdash (ts)^\rho} \qquad (\multimap^-)$$

$$\frac{\Gamma \vdash t^{\tau \times \rho}}{\Gamma \vdash (t\mathbf{t})^\tau} \qquad (\times_1^-)$$

$$\frac{\Gamma \vdash t^{\tau \times \rho}}{\Gamma \vdash (t\mathbf{f})^\rho} \qquad (\times_0^-)$$

$$\frac{\Gamma_1 \vdash t^{\mathrm{bool}} \qquad \Gamma_2 \vdash s^\tau \qquad \Gamma_2 \vdash r^\tau}{\Gamma_1, \Gamma_2 \vdash (t\,\langle s, r\rangle)^\tau} \qquad (\mathrm{bool}^-)$$

$$\frac{\Gamma_1 \vdash t^{\tau \otimes \rho} \qquad \Gamma_2, x^\tau, y^\rho \vdash s^\sigma}{\Gamma_1, \Gamma_2 \vdash (t(\lambda x^\tau, y^\rho\, s))^\sigma} \qquad (\otimes^-)$$

$$\frac{\Gamma \vdash t^{\mathbf{N}} \qquad \emptyset \vdash {s_0}^{\diamond \multimap \tau \multimap \tau} \qquad \emptyset \vdash {s_1}^{\diamond \multimap \tau \multimap \tau}}{\Gamma \vdash (t\,\{s_0, s_1\})^{\tau \multimap \tau}} \qquad (\mathbf{N}^-)$$

$$\frac{\Gamma \vdash t^{\mathbf{L}(\tau)} \qquad \emptyset \vdash s^{\diamond \multimap \tau \multimap \rho \multimap \rho}}{\Gamma \vdash (t\,\{s\})^{\rho \multimap \rho}} \qquad (\mathbf{L}(\tau)^-)$$

Aus obiger Definition unmittelbar ersichtlich ist

[7] Diese in der Informatik übliche und im folgenden stillschweigend verwendete Art zur Aufschreibung strikt positiver induktiver Definitionen ist so zu verstehen, daß die Relation $\cdot \vdash \cdot$ die kleinste Relation ist, die unter den angegebenen Regeln abgeschlossen ist. Dabei stehen bei jeder Regel die Voraussetzungen oberhalb und die Konklusion unterhalb der waagerechten Linie. (Var) etwa besagt, daß stets $\Gamma, x^\tau \vdash x^\tau$ gelten soll. $(\times^+)$ besagt: "Wenn $\Gamma \vdash t^\tau$ und $\Gamma \vdash s^\rho$ gilt, so auch $\Gamma \vdash \langle t, s\rangle^{\tau\times\rho}$."

Proposition 1.1.6 (weakening). $\Gamma \vdash t^\tau; \Gamma \subset \Gamma' \Longrightarrow \Gamma' \vdash t^\tau$

Die Schreibweise t^τ soll t bezeichnen, aber nur gebildet werden können, wenn es ein Γ gibt mit $\Gamma \vdash t^\tau$. Ein solches τ soll auch "der Typ von t" heißen. Daß diese Definition sinnvoll ist, zeigt folgendes

Lemma 1.1.7. *Für jeden Term t gibt es höchstens einen Typ τ mit t^τ. In diesem Fall gibt es ein kleinstes Γ mit $\Gamma \vdash t^\tau$.*

Beweis. Induktion(t) □

Das durch die Eigenschaft "kleinst" eindeutig bestimmte Γ heißt auch "Menge der freien Variablen".

Lemma 1.1.8 (Kopfform). *Sei $\Gamma \vdash t^\tau$. Dann ist t von einer der Formen $x\vec{t}$, $c\vec{t}$, $(\lambda x^\rho s)\vec{t}$ oder $\langle s, r\rangle \vec{t}$.*

Beweis. Induktion(t) unter Beachtung, daß es keine getypten Terme der Form $\{\cdot\}$ und $\{\cdot,\cdot\}$ gibt. □

1.2 Reduktionen

Reduktionen können wie üblich vorgenommen werden, wenn eine Beseitigung unmittelbar auf eine "Einführung" trifft. Bei Einfürung durch Konstanten wird dabei Wert darauf gelegt, daß das Objekt "vollständig eingeführt" ist, also als Konstruktorterm gegeben ist. Proposition 1.2.8 wird zeigen, daß die Reduktionsregeln und die gewählte Datenrepräsentation (als Konstruktorterme) zusammenpassen.

Definition 1.2.1 (Numerale, Listen). *Numerale* sind Terme der Form $\mathbf{S}_{i_1} d_1^\diamond(\ldots(\mathbf{S}_{i_n} d_n^\diamond 0))$. *Listen* sind Terme der Gestalt $\mathbf{cons}_\tau d_1^\diamond a_1^\tau(\ldots(\mathbf{cons}_\tau d_n^\diamond a_n^\tau \mathbf{nil}_\tau))$. Ein Numeral bzw. eine Liste heißt *Eingabenumeral* bzw. *Eingabeliste*, falls die $\vec{d}$ sämtlich Variablen sind.

Im folgenden seien $d, \ldots$ Mitteilungszeichen für getypte Terme vom Typ $\diamond$ und ν Mitteilungszeichen für Numerale sowie ℓ Mitteilungszeichen für Listen.

Definition 1.2.2 (elementare Reduktionen). Die Relation $\mapsto$ wird definiert durch:

$(\lambda x^\tau\, t)s \mapsto t[s/x]$
$\langle t, s\rangle\, \mathbf{t} \mapsto t$
$\langle t, s\rangle\, \mathbf{f} \mapsto s$
$\mathbf{t}\, \langle t, s\rangle \mapsto t$
$\mathbf{f}\, \langle t, s\rangle \mapsto s$
$\otimes_{\tau,\rho} ts(\lambda x^\tau, y^\rho\, r) \mapsto r[t, s/x, y]$
$0\, \{t_0, t_1\}\, s \mapsto s$
$\mathbf{S}_i d^\diamond \nu\, \{t_0, t_1\}\, s \mapsto t_i d^\diamond(\nu\, \{t_0, t_1\}\, s)$ ν ein Numeral
$\mathbf{nil}_\tau\, \{t\}\, s \mapsto s$
$\mathbf{cons}_\tau d^\diamond a \ell\, \{t\}\, s \mapsto t d^\diamond a(\ell\, \{t\}\, s)$ ℓ eine Liste

Dabei bezeichnet $t[s/x]$ wie üblich die t nach Substitution aller (freien) Vorkommen von x durch s. Diese Substitution ist als "capture free substitution" zu verstehen, d.h. es ist bei der Berechnung von $(\lambda xt)[s/y]$ zu $\lambda x(t[s/y])$ stets von einer Umbenennung von λxt auszugehen, bei der x von y und den freien Variablen von s verschieden ist. Da Terme identifiziert werden, die sich nur durch Umbenennung gebundener Variablen unterscheiden, ist dies wohldefiniert. Ferner bezeichnet $t[r, s/x, y]$ die *simultane* Ersetzung von x und y in t durch r und s.

Bemerkung 1.2.3. Die Reduktionen für $\mathbf{N}$ und $\otimes$ sind die gleichen wie im ungetypen Lambda-Kalkül. Genauer besagen diese Regeln, daß sich Numerale verhalten wie Churchsche Ziffern und $\otimes_{\tau,\rho}$ wie der Paarbildungsoperator, also wie $\lambda x, y, z\, zxy$.

Die folgende Definition des Termabschlusses geht von der Vorstellung aus, daß zwar innerhalb von "Daten" (also $\mathbf{S}_i t$, $\otimes_{\tau,\rho} st$, ...), nicht aber unter Abstraktionen (also λ-Abstraktion $\lambda x\, t$ und Paarbildung[8] $\langle s,t\rangle$) reduziert werden darf. Mit der gewählten Notation läßt sich dies besonders einfach ausdrücken:

Definition 1.2.4 ($t \to t'$). Die Relation $t \to t'$ wird induktiv definiert durch:

$$\frac{t \mapsto t'}{t \to t'} \qquad \frac{t \to t'}{ts \to\ t's} \qquad \frac{s \to s'}{ts \to\ ts'}$$

Lemma 1.2.5. $\Gamma_1 \cup \{x^\rho\} \vdash t^\tau$*;* $\Gamma_2 \vdash s^\rho$*;* Γ_1, Γ_2 *disjunkt* $\Longrightarrow \Gamma_1, \Gamma_2 \vdash t[s/x^\rho]^\tau$

Beweis. Induktion(t)

y Falls $y \neq x$, so ist $y[s/x] = y$. Nach Voraussetzung ist $\Gamma_1 \vdash y^\tau$, also (Proposition 1.1.6) auch $\Gamma_1, \Gamma_2 \vdash y^\tau$

Falls $y = x$, so $y[s/x] = s$. Nach Voraussetzung $\Gamma_2 \vdash s^\rho$, also (Proposition 1.1.6) auch $\Gamma_1, \Gamma_2 \vdash s^\rho$.

c klar

$\lambda y^\sigma.t'$ Dann ist τ von der Form $\tau = \sigma \multimap \tau'$.

Ohne Einschränkung kann angenommen werden, daß y hinreichend neu ist. Wegen $\Gamma_1 \vdash (\lambda y^\sigma.t')^{\sigma\multimap\tau'}$ ist $\Gamma_1, y^\sigma \vdash t'^{\tau'}$, nach Induktionsvoraussetzung also $\Gamma_1, \Gamma_2, y^\sigma \vdash (t'[s/x])^{\tau'}$, also $\Gamma_1, \Gamma_2 \vdash \underbrace{(\lambda y^\sigma\ (t'[s/x]))}_{(\lambda y^\sigma.t')[s/x]}{}^{\sigma\multimap\tau'}$.

$\langle r_1, r_2\rangle$ folgt unmittelbar aus der Induktionsvoraussetzung.

$r_1 r_2$ Dann ist(!) r_2 geschlossen oder es gibt disjunkte Γ_{11}, Γ_{12} mit $\Gamma_{11} \vdash {r_1}^{\tau'}$ und $\Gamma_{12} \vdash {r_2}^{\tau''}$. In jedem Fall zieht sich die Substitution nur in einen Teilterm hinein, so daß die Behauptung unmittelbar aus der Induktionsvoraussetzung folgt.

[8]Paarbildung kann (im Gegensatz zum Tensor) als Abstraktion aufgefaßt werden, da das Paar (genau wie eine durch λ–Abstraktion gebildete Funktion) mehr "Information" enthält als bei einer *einmaligen* Verwendung zurückgewonnen werden kann.

$\{r\}$, $\{r_1, r_2\}$ Diese Fälle scheiden aus, da es keine getypten Terme dieser Bauart gibt.

$\square$

Lemma 1.2.6 (Abschluß unter Reduktion). $\Gamma \vdash t^\tau ;\ t \to t' \Longrightarrow \Gamma \vdash t'^\tau$

Beweis. Induktion(t). Es müssen also nur elementare Reduktionen betrachtet werden. Für diese ist die Behauptung aber offenbar richtig bzw. folgt aus Lemma 1.2.5. $\square$

Da die "Größe" eines Terms durch die Zahl der freien Variablen wiedergegeben werden soll, sind geschlossene Terme nicht von besonderem Interesse. An ihre Stelle treten "fast geschlossene" Terme:

Definition 1.2.7 (fast geschlossen). Ein Term t heißt *fast geschlossen*, falls er getypt ist und alle freien Variablen den Typ $\diamond$ haben; mit anderen Worten, falls es ein Γ gibt mit $\Gamma \vdash t^\tau$ und $x \in \Gamma \Longrightarrow x^\diamond$.

Damit kann jetzt gezeigt werden, daß das Termsystem "ausreichend viele" Reduktionsregeln enthält, daß die Normalformen von "Datentypen" also wirklich das sind, was man sich unter einem Ergebnis vorstellt.

Proposition 1.2.8. *Jeder normale, fast geschlossene Term vom Typ* $\mathbf{N}$ *ist ein Numeral. Jeder normale, fast geschlossene Term vom Typ* $\mathbf{L}(\tau)$ *ist eine Liste. Jeder normale, fast geschlossene Term vom Typ* bool *ist* $\mathbf{t}$ *oder* $\mathbf{f}$. *Jeder normale, fast geschlossene Term vom Typ* $\diamond$ *ist eine Variable.*

Beweis. Sei also t getypt von einem der Typen $\mathbf{N}$, $\mathbf{L}(\tau)$, bool, oder $\diamond$.

Induktion(t). Fallunterscheidung gemäß Lemma 1.1.8.

$x\vec{t}$ Da t fast geschlossen ist $x^\diamond$, also $\|\vec{t}\| = 0$, da es keine Beseitigungsregel für den Typ $\diamond$ gibt. Insbesondere ist der Term also eine Variable.

$c\vec{t}$ Es sind folgende Fälle zu betrachten:

- Ist $c \in \{\mathbf{t}, \mathbf{f}\}$, so ist $\|\vec{t}\| = 0$, da der Term sonst nicht normal wäre (und der Term ist $\mathbf{t}$ oder $\mathbf{f}$).
- Ist $c \in \{0, \mathbf{nil}_\rho\}$, so ist der Term im Falle $\|\vec{t}\| \geq 2$ nicht normal, im Falle $\|\vec{t}\| = 1$ nicht vom richtigen Typ und im Falle $\|\vec{t}\| = 0$ tatsächlich ein Numeral bzw. eine Liste.
- Ist $c \in \{\mathbf{S}_0, \mathbf{S}_1\}$, so muß $\|\vec{t}\| \geq 2$ sein (da der Term sonst den falschen Typ hätte). Nach Induktionsvoraussetzung ist t_2 dann Numeral. Also muß $\|\vec{t}\| = 2$ sein (da der Term sonst nicht normal wäre) und der Term ist also tatsächlich ein Numeral.
- Ist $c = \mathbf{cons}_\rho$, so muß $\|\vec{t}\| \geq 3$ sein, da der Term sonst $\multimap$-Typ hätte. Nach Induktionsvoraussetzung ist t_3 eine Liste. Damit $\|\vec{t}\| = 3$, da der Term sonst nicht normal wäre. Insbesondere ist der Term eine Liste.

– Ist $c = \otimes_{\rho,\sigma}$, so ist der Term im Falle $\|\vec{t}\| \leq 2$ nicht vom richtigen Typ und im Falle $\|\vec{t}\| \geq 3$ nicht normal.

$(\lambda x^\rho\, s)\vec{t}$ Ist $\|\vec{t}\| \geq 1$, so ist der Term nicht normal, andernfalls nicht vom richtigen Typ.

$\langle s, t\rangle\, \vec{t}$ ebenso.

∎

Bemerkung 1.2.9. Der Beweis ergibt sogar, daß die normalen, fast geschlossenen Terme vom Typ $\mathbf{N}$ bzw. $\mathbf{L}(\cdot)$ sogar *Eingabe*numerale bzw. *Eingabe*listen sind. Es werden also, auch wenn das nicht nötig wäre (da dem Typ $\diamond$ ohnehin keine semantische Bedeutung beigemessen wird), für Ein- und Ausgabe das gleiche "Format" verwendet.

1.3 Länge von Reduktionsketten

Definition 1.3.1 (Länge eines Terms). Die Länge $|t|$ eines Terms t wird durch Induktion über den Aufbau definiert:

$|c| = |x| = 1$
$|ts| = |t| + |s|$
$|\lambda x^\tau\, s| = |s| + 1$
$|\langle t, s\rangle| = \max\{|t|, |s|\} + 1$
$|\{t\}| = |\{t_1, t_2\}| = 0$

Außerdem wird die Schreibweise $|\vec{t}|$ für $\sum_{i=1}^{n} |t_i|$ verwendet.

Die lineare Typisierung hat zur Folge, daß in jedem typisierten Term jede Variable "im wesentlichen" höchstens einmal verwendet wird. Insbesondere werden durch Substitution Terme nicht verdoppelt, so daß die Länge nicht zunimmt. Genauer gilt folgendes

Lemma 1.3.2. $\Gamma_1 \vdash t^\tau$*;* $\Gamma_2 \vdash s^\rho$*;* Γ_1, Γ_2 *disjunkt* $\Longrightarrow |t[s/x]| \leq |t| + |s|$

Insbesondere ist $|(\lambda x^\tau\, t)s| > |t[s/x]|$.

Beweis. Induktion(t). Die Fälle c und x sind unmittelbar klar; in den Fällen $\lambda x^\rho\, t'$ und $\langle r_1, r_2\rangle$ folgt die Behauptung jeweils unmittelbar aus den entsprechenden Induktionsvoraussetzungen; im Fall $r_1 r_2$ folgt aus der Typisiertheit, daß die freien Variablen von r_1 und r_2 disjunkt sind; die Fälle $\{r\}$ und $\{r_1, r_2\}$ treten wegen der Typisiertheit nicht auf. ∎

Lemma 1.3.3. $\Gamma \vdash t^\diamond \Longrightarrow \Gamma \neq \emptyset$

Beweis. Induktion($|t|$). Fallunterscheidung gemäß Lemma 1.1.8.

$x\vec{t}$ Dann ist $x \in \Gamma$

$c\vec{t}$ Es sind folgende Fälle zu unterscheiden:

- Ist $c \in \{\mathbf{t}, \mathbf{f}\}$, so muß wegen $\Gamma \vdash t^\diamond$ gelten daß $\|\vec{t}\| \geq 1$ ist; t_1 ist damit von der Form $t_1 = \langle t_1', t_1'' \rangle$. Anwendung der Induktionsvoraussetzung auf den Term $t_1' t_2 \dots t_n$ liefert die Behauptung.
- Ist $c \in \{0, \mathbf{nil}_\tau\}$, so folgt $\|\vec{t}\| \geq 2$ und die Induktionsvoraussetzung kann auf $t_2 t_3 \dots t_n$ angewendet werden.
- Ist $c \in \{\mathbf{S}_0, \mathbf{S}_1, \mathbf{cons}_\tau\}$, so folgt $\|\vec{t}\| \geq 1$ und die Induktionsvoraussetzung kann auf t_1 angewendet werden.
- Ist $c = \otimes_{\tau,\rho}$, so muß $\|\vec{t}\| \geq 3$ sein. t_3 ist dann von der Form $t_3 = \lambda x, y.t_3'$ und die Induktionsvoraussetzung kann auf den kürzeren (Lemma 1.3.2) Term $t_3'[t_1, t_2/x, y]$ angewendet werden.

$(\lambda x.s)\vec{t}$ Dann ist $\|\vec{t}\| \geq 1$ und die Induktionshypothese kann auf $s[t_1/x]t_2 \dots t_n$ angewendet werden (nach Lemma 1.3.2).

$\langle r, s \rangle \vec{t}$ Dann ist $\|\vec{t}\| \geq 1$ und die Induktionshypothese kann auf $rt_2 \dots t_n$ angewandt werden.

□

Bemerkung 1.3.4. Betrachtet man obigen Beweis, so stellt man fest, daß nur konstruktive Schlüsse verwandt wurden. Es wurde also sogar (die vom intuitionistischen Standpunkt aus stärkere Aussage) "$\Gamma \vdash t^\diamond$ dann besitzt Γ ein Element" gezeigt. Ferner stellt man fest, daß der sich aus dem Beweis ergebende Algorithmus zum Finden einer freien Variable mit linear vielen "Schritten" auskommt, von denen keiner aufwendiger ist als die Berechnung einer Substitution. Insbesondere ist der Beweis also "feasible" (d.h. der Beweis ist konstruktiv und der extrahierte Algorithmus liegt in `Ptime`).

Korollar 1.3.5 (non sizeincreasing property). *Jede durch einen geschlossenen Term in $\mathcal{C}$ definierbare Funktion vom Typ $\vec{\mathbf{N}} \multimap \mathbf{N}$ hat die Eigenschaft, daß die Länge der Ausgabe die Gesamtlänge der Eingabe nicht übersteigt. Dabei ist unter Länge die übliche Länge, also die Zahl der Ziffern der Dualdarstellung zu verstehen.*

Beweis. Sei $\emptyset \vdash t^{\vec{\mathbf{N}} \multimap \mathbf{N}}$ und $\vec{\nu}$ Eingabenumerale. Dann ist wegen der linearen Typisierung $|\mathrm{FV}(\vec{\nu})|$ gerade die Zahl der Ziffern der Dualdarstellung. Wegen Proposition 1.2.8 ist die Normalform von $t\vec{\nu}$ ein Numeral. Die Zahl der Ziffern dieses Numerals ist beschränkt durch die Zahl der Teilterme vom Typ $\diamond$, diese wegen obigen Lemma und der linearen Typisierung durch die Zahl der freien Variablen. □

Bemerkung 1.3.6. Akzeptiert man das Maß "Zahl der freien Variablen" als Längenmaß auch für Terme höheren Typs, so gilt obiges Korollar trivialerweise für *beliebige* Typen (und nicht nur für $\vec{\mathbf{N}} \multimap \mathbf{N}$).

$\mathbb{N}[X]$ bezeichne die Menge der Polynome in einer Unbestimmten X über den natürlichen Zahlen. Als Mitteilungszeichen für Polynome werden $f, g, \dots$ verwendet.

Definition 1.3.7 (natürliche Ordnung auf $\mathbb{N}[X]$). Auf $\mathbb{N}[X]$ wird durch

$$\sum a_i X^i \preccurlyeq \sum b_i X^i \iff \forall i . a_i \leq b_i$$

die teilweise Ordnung $\preccurlyeq$ definiert.

Bezüglich dieser Ordnung existieren endliche Suprema[9]. Diese werden mit $\sup_{\preccurlyeq}\{\ldots\}$ bezeichnet.

Proposition 1.3.8. *$f, g \in \mathbb{N}[X]$; $f \preccurlyeq g$; $n \in \mathbb{N} \Longrightarrow f(n) \leq g(n)$*

Definition 1.3.9 (Polynomiale Länge $\vartheta(t)$ eines Terms t). Für jeden Term t wird durch Induktion über den Aufbau ein Polynom $\vartheta(t) \in \mathbb{N}[X]$ definiert.

$\vartheta(x) = \vartheta(c) = 0$
$\vartheta(ts) = \vartheta(t) + \vartheta(s)$
$\vartheta(\lambda x^\tau\, t) = \vartheta(t)$
$\vartheta(\langle t, s\rangle) = \sup_{\preccurlyeq}\{t, s\}$
$\vartheta(\{t\}) = X \cdot \vartheta(t) + X \cdot |t|$
$\vartheta(\{t, s\}) = X \cdot \sup_{\preccurlyeq}\{\vartheta(t), \vartheta(s)\} + X \cdot \max\{|t|, |s|\}$

Außerdem wird die Schreibweise $\vartheta(\vec{t})$ für $\sum_{i=1}^{n} \vartheta(t_i)$ verwendet.

Lemma 1.3.10. *$\Gamma_1 \vdash t^\tau, \Gamma_2 \vdash s^\rho$; Γ_1, Γ_2 disjunkt $\Longrightarrow \vartheta(t[s/x]) \preccurlyeq \vartheta(t) + \vartheta(s)$*

Beweis. Induktion(t). Analog zum Beweis von 1.3.2. ∎

Der folgende Satz, der die Existenz einer polynomial kurzen Reduktionsfolge zusichert, geht von einer speziellen Reduktionsstrategie aus. Diese ist dadurch gekennzeichnet, daß Iterationen *sofort* ausgefaltet werden; so wird etwa $\mathbf{S}_{i_1} r_1^\diamond(\ldots(\mathbf{S}_{i_n} r_n^\diamond 0))\{h_0, h_1\}\, t$ zunächst zu $h_{i_1} r_1^\diamond(\ldots(h_{i_n} r_n^\diamond t))$ reduziert *bevor* sich eventuell neu ergebende Redexe bearbeitet werden. Man beachte, daß die geschweiften Klammern verschwunden sind zugunsten einer n–fachen Wiederholung eines der Terme h_0 oder h_1. Da n durch die Zahl der freien Variablen des Terms beschränkt wird, verwundert es nicht, das $\vartheta(t)(|\mathrm{FV}(t)|)$ eine obere Schranke für die Zahl der Reduktionsschritte bis zur Normalform ist. Aus technischen Gründen ist es aber notwendig, bei der Formulierung der Aussage nichts zu verschenken und das in der Normalform steckende "Potential" mit zu berücksichtigen:

Satz 1.3.11. *$\Gamma \vdash t^\tau$; $\natural \in \mathbb{N}$, $\natural \geq |\mathrm{FV}(t)|$. Dann gibt es $t = t_0 \to t_1 \to \ldots \to t_k$ mit t_k normal und $\vartheta(t_k)(\natural) + |t_k| + k \leq \vartheta(t)(\natural) + |t|$*

Beweis. Induktion($\vartheta(t)(\natural) + |t|$). Die Fallunterscheidung wird nach dem Aufbau von t gemäß Lemma 1.1.8 erfolgen. Zunächst werden jedoch die interessanten Fälle in allgemeinerer Form betrachtet:

[9] Das Supremum zu $\sum a_i X^i$ und $\sum b_i X^i$ ist $\sum \max\{a_i, b_i\} X^i$.

$t = r^{\mathbf{N}}\{h_0, h_1\}\, s\vec{t}$**.** Normalisiere r nach Induktionshypothese mit k_0 Schritten zu r', wobei

$$\vartheta(r')(\natural) + |r'| + k_0 \leq \vartheta(r)(\natural) + |r|$$

Ohne Einschränkung kann angenommen werden, daß r' ein Numeral ist (anderenfalls sind nur noch s und $\vec{t}$ zu normalisieren, was nach Induktionshypothese in ausreichend geringer Zahl von Schritten geht), also $r' = \mathbf{S}_{i_1} r_1^{\diamond}(\ldots(\mathbf{S}_{i_n} r_n^{\diamond} 0))$. Wegen Lemma 1.3.3 und den linearen Typisierungsregeln ist $n \leq \natural$. Bilde in $n+1$ Schritten $t' = h_{i_1} r_1^{\diamond}(\ldots(h_{i_n} r_n^{\diamond} s))\vec{t}$. Normalisiere nach Induktionshypothese t' zu t'' mit k_1 Schritten, wobei

$$\vartheta(t'')(\natural) + |t''| + k_1 \leq \vartheta(t')(\natural) + |t'|$$

Es ist $\vartheta(t') \preccurlyeq n \cdot \sup_{\preccurlyeq}\{\vartheta(h_0), \vartheta(h_1)\} + \vartheta(\vec{r}) + \vartheta(s) + \vartheta(\vec{t})$ und $|t'| \leq n \cdot \max\{|h_0|, |h_1|\} + |\vec{r}| + |s| + |\vec{t}|$. Insgesamt ergibt sich also

$$\begin{array}{ll}
 & \vartheta(t'')(\natural) + |t''| + k_1 + n + 1 + k_0 \\
\leq & \vartheta(t')(\natural) + |t'| + n + 1 + k_0 \\
\leq & n \cdot \sup_{\preccurlyeq}\{\vartheta(h_0), \vartheta(h_1)\}(\natural) + \vartheta(\vec{r})(\natural) + \vartheta(s)(\natural) + \vartheta(\vec{t})(\natural) \\
 & + n \cdot \max\{|h_0|, |h_1|\} + |\vec{r}| + |s| + |\vec{t}| \\
 & + n + 1 + k_0 \\
= & \underbrace{|\vec{r}| + n + 1}_{|r'|} + k_0 + \underbrace{\vartheta(\vec{r})}_{\vartheta(r')}(\natural) + |s| + |\vec{t}| \\
 & + \vartheta(s)(\natural) + \vartheta(\vec{t})(\natural) + \\
 & n \cdot \sup_{\preccurlyeq}\{\vartheta(h_0), \vartheta(h_1)\}(\natural) + n \cdot \max\{|h_0|, |h_1|\} \\
\leq & \vartheta(r)(\natural) + \underbrace{|r| + |s| + |\vec{t}|}_{|t|} \\
 & + \vartheta(s)(\natural) + \vartheta(\vec{t})(\natural) \\
 & + n \cdot \sup_{\preccurlyeq}\{\vartheta(h_0), \vartheta(h_1)\}(\natural) + n \cdot \max\{|h_0|, |h_1|\} \\
\leq & \vartheta(t)(\natural) + |t|
\end{array}$$

$t = r^{\mathbf{L}(\tau)}\{h\}\, s\vec{t}$**.** Normalisiere nach Induktionshypothese r zu r' in k_0 Schritten mit

$$\vartheta(r')(\natural) + |r'| + k_0 \leq \vartheta(r)(\natural) + |r|$$

Ohne Einschränkung ist r' eine Liste, also $r' = \mathbf{cons}_\tau d_1^{\diamond} r_1^{\tau}(\ldots(\mathbf{cons}_\tau d_n^{\diamond} r_n^{\tau} \mathbf{nil}_\tau))$. Wegen Lemma 1.3.3 ist $n \leq \natural$. Bilde in $n+1$ Schritten den Term $t' = h d_1^{\diamond} r_1^{\tau}(\ldots(h d_n^{\diamond} r_n^{\tau} s))\vec{t}$. Normalisiere t' nach Induktionshypothese zu t'' in k_1 Schritten mit

$$k_1 + \vartheta(t'')(\natural) + |t''| \leq \vartheta(t')(\natural) + |t'|$$

Es ist $\vartheta(t') = n\vartheta(h) + \vartheta(\vec{d}) + \vartheta(\vec{r}) + \vartheta(s) + \vartheta(\vec{t})$ und $|t'| = n|h| + |\vec{d}| + |\vec{r}| + |s| + |\vec{t}|$.

Insgesamt ergibt sich also:

$$\begin{array}{rl}
& \vartheta\left(t''\right)(\natural) + |t''| + k_1 + n + 1 + k_0 \\
\leq & \vartheta\left(t'\right)(\natural) + |t'| + n + 1 + k_0 \\
= & n\vartheta\left(h\right)(\natural) + \vartheta\left(\vec{d}\right)(\natural) + \vartheta\left(\vec{r}\right)(\natural) + \vartheta\left(s\right)(\natural) + \vartheta\left(\vec{t}\right)(\natural) \\
& +n\left|h\right| + \left|\vec{d}\right| + \left|\vec{r}\right| + \left|s\right| + \left|\vec{t}\right| + n + 1 + k_0 \\
= & \underbrace{\vartheta\left(\vec{d}\right)(\natural) + \vartheta\left(\vec{r}\right)(\natural)}_{\vartheta(r')(\natural)} + \underbrace{\left|\vec{d}\right| + \left|\vec{r}\right| + n + 1}_{|r'|} + k_0 \\
& +n\vartheta\left(h\right)(\natural) + n\left|h\right| + \vartheta\left(s\right)(\natural) + \vartheta\left(\vec{t}\right)(\natural) + \left|s\right| + \left|\vec{t}\right| \\
\leq & \vartheta\left(r\right)(\natural) + \left|r\right| \\
& +n\vartheta\left(h\right)(n) + n\left|h\right| + \vartheta\left(s\right)(\natural) + \vartheta\left(\vec{t}\right)(\natural) + \left|s\right| + \left|\vec{t}\right| \\
\leq & \vartheta\left(t\right)(\natural) + \left|t\right|
\end{array}$$

$t = r^{\tau \multimap \rho} s^\tau \vec{t}$**.** Normalisiere nach Induktionshypothese r zu r' in k_0 Schritten mit

$$\vartheta\left(r'\right) + |r'| + k_0 \leq \vartheta\left(r\right)(\natural) + |r|$$

Ohne Einschränkung kann angenommen werden, daß r' von der Form $\lambda x^\tau r''$ ist (andernfalls klar, da sonst nur noch s und $\vec{t}$ normalisiert werden müssen). Bilde in einem Schritt $t' = r''[s/x]\vec{t}$ und normalisiere nach Induktionshypothese t' in k_2 Schritten zu t'', wobei

$$\vartheta\left(t''\right)(\natural) + |t''| + k_1 \leq \vartheta\left(t'\right)(\natural) + |t'|$$

Wegen $|t'| = |r''[s/x]| + \left|\vec{t}\right| \overset{1.3.2}{\leq} |r''| + |s| + \left|\vec{t}\right|$ und $\vartheta\left(t'\right) = \vartheta\left(r''[s/x]\right) + \vartheta\left(\vec{t}\right) \overset{1.3.10}{\precsim} \vartheta\left(r''\right) + \vartheta\left(s\right) + \vartheta\left(\vec{t}\right)$ ergibt sich

$$\begin{array}{rl}
& \vartheta\left(t''\right)(\natural) + |t''| + k_1 + k_0 + 1 \\
\leq & \vartheta\left(t'\right)(\natural) + |t'| + k_0 + 1 \\
\leq & \vartheta\left(r''\right)(\natural) + \vartheta\left(s\right)(\natural) + \vartheta\left(\vec{t}\right)(\natural) + |r''| + |s| + \left|\vec{t}\right| + k_0 + 1 \\
= & \underbrace{\vartheta\left(r''\right)}_{\vartheta(r')}(\natural) + \underbrace{|r''| + 1}_{|r'|} + k_0 + \vartheta\left(s\right)(\natural) + \vartheta\left(\vec{t}\right)(\natural) + |s| + \left|\vec{t}\right| \\
\leq & \vartheta\left(r\right)(\natural) + |r| + \vartheta\left(s\right)(\natural) + \vartheta\left(\vec{t}\right)(\natural) + |s| + \left|\vec{t}\right| \\
= & \vartheta\left(t\right)(\natural) + |t|
\end{array}$$

Hier nun also die eigentliche Fallunterscheidung nach dem Aufbau von t:

$x\vec{t}$ Hier sind nur die $\vec{t}$ zu normalisieren, was nach Induktionsvoraussetzung ausreichend schnell geht.

$c\vec{t}$ Für $c \in \{0, \mathbf{S}_0, \mathbf{S}_1, \mathbf{nil}_\rho, \mathbf{cons}_\rho\}$ folgt die Behauptung entweder unmittelbar aus den Induktionsvoraussetzungen ($\vec{t}$ ausreichend kurz) oder es handelt sich um Instanzen der oben diskutierten Fälle.

Für $c \in \{\mathbf{t}, \mathbf{f}\}$ ist der Term normal oder t_2 ist von der Form $\langle t_2', t_2'' \rangle$. Dies reduziert in einem Schritt zu $t_2' t_3 \dots t_n$ beziehungsweise zu $t_2'' t_3 \dots t_n$ und die Induktionsvoraussetzung liefert den Rest.

Für $c = \otimes_{\rho,\sigma}$ ist nur $\|\vec{t}\| \geq 3$ interessant. In diesem Fall ist t_3 von der Form $t_3 = \lambda x, y.t_3'$, der Term reduziert in einem Schritt zu $t_3'[t_1, t_2/x, y]t_4 \dots t_n$ und analog zum Fall $r^{\tau \multimap \rho} s^\tau \vec{t}$ liefert die Induktionsvoraussetzung den Rest.

$(\lambda xs)\vec{t}$ normal ($\|\vec{t}\| = 0$) oder Instanz eines bereits betrachteten Falls.

$\langle s, r \rangle \vec{t}$ Nicht normal nur im Fall $\|\vec{t}\| \geq 1$. Dann ist $t_1 \in \{\mathbf{t}, \mathbf{f}\}$, der Term reduziert in einem Schritt zu $st_2 \dots t_n$ beziehungsweise zu $rt_2 \dots t_n$ und die Induktionsvoraussetzung liefert den Rest.

□

Korollar 1.3.12 (Hofmann). $\emptyset \vdash t^{\vec{\mathbf{N}} \multimap \mathbf{N}}$. *Dann bezeichnet t eine Funktion, die in polynomial vielen Schritten berechnet werden kann.*

Beweis. Seien $\vec{\nu}$ Eingabenumerale. Dann ist $\vartheta(t\vec{\nu}) = \vartheta(t)$, da $\vartheta(\vec{\nu}) = 0$. Eine obere Schranke für die Zahl der Schritte ist also $\vartheta(t)(|\mathrm{FV}(\vec{\nu})|) + |t| + |\vec{\nu}|$ und damit polynomial in der Länge der Eingabe $\vec{\nu}$. □

1.4 Beispiel

Beispiel 1.4.1 (Insertion sort). Sei τ ein Typ, auf dem eine lineare Ordnung definiert ist. Diese möge durch einen Term $\lhd_\tau$ vom Typ $\tau \multimap \tau \multimap \mathrm{bool} \otimes (\tau \otimes \tau)$ berechnet werden, das heißt $\lhd_\tau ts \to^* \otimes_{\mathrm{bool},\tau\otimes\tau} \mathbf{t}(\otimes_{\tau,\tau} ts)$, falls s "kleiner" als t und $\lhd_\tau ts \to^* \otimes_{\mathrm{bool},\tau\otimes\tau} \mathbf{f}(\otimes_{\tau,\tau} ts)$ sonst[10].

Ein Element a kann in eine aufsteigend sortierte Liste vom Typ $\mathbf{L}(\tau)$ wie folgt eingeordnet werden:

> Bei der leeren Liste ist es klar. Bei einer längeren Liste vergleiche a mit dem ersten Element der Liste. Ist a kleiner, so hänge a vorne an die Liste an, ansonsten füge a in den Rest der Liste ein.

Betrachtet man diesen Algorithmus, so fällt auf, daß in jedem Schritt wahlweise auf den Rest der Liste oder einen rekursiven Aufruf ("füge a in den Rest der Liste ein") zurückgegriffen wird. Dieses "entweder–oder" ist im Sinne der linearen Typisierung auch nötig, da beides Ausgaben im Umfang der gesamten Liste produzieren kann.

[10] Auch wenn es zunächst unmöglich oder zumindest umständlich erscheint, "interessante" Ordnungen in diesem Termsystem zu definieren, so ist dieses Beispiel durchaus von Interesse: man kann nämlich einfach $\lhd_\tau$ als neue Konstante mit den entsprechenden Reduktionsregeln hinzunehmen. Wenn man nun die Definition von $|t|$ dahingehend abändert, daß $|\lhd_\tau| = 4$ ist, so gelten alle Sätze weiterhin und man erhält eine Schranke für die Zahl der "Unterprogrammaufrufe" von $\lhd_\tau$, unabhängig davon wie komplex die Ordnung tatsächlich ist.

Um eine Einfügefunktion vom Typ $\mathbf{L}(\tau) \multimap \diamond \multimap \tau \multimap \mathbf{L}(\tau)$ zu definieren[11], wird also durch Iteration über das erste Argument ein Paar (im Sinne von $\times$) bestehend aus einem Einfügefunktional vom Typ $\diamond \multimap \tau \multimap \mathbf{L}(\tau)$ ("füge in den Rest der Liste ein") und der Liste selbst definiert. Es ist also:

$$\begin{aligned}
&\texttt{insert} = \lambda l^{\mathbf{L}(\tau)}\, l\{\lambda z_1^{\diamond}, p_1^{\tau}, x^{(\diamond\multimap\tau\multimap\mathbf{L}(\tau))\times\mathbf{L}(\tau)} \\
&\langle \lambda z_2^{\diamond}, p_2^{\tau} \lhd_\tau p_1 p_2 (\lambda y^{\mathrm{bool}}, p^{\tau\otimes\tau}\, y\langle \quad p(\lambda p_1^{\tau}, p_2^{\tau}\, \mathbf{cons}_\tau z_2 p_2(\mathbf{cons}_\tau z_1 p_1 (x\mathbf{f}))), \\
&\qquad\qquad\qquad\qquad\qquad\qquad\quad p(\lambda p_1^{\tau}\, p_2^{\tau}\, \mathbf{cons}_\tau z_1 p_1 (x\mathbf{t} z_2 p_2))\rangle, \\
&\mathbf{cons}_\tau z_1 p_1 (x\mathbf{f})\rangle\} \,\langle \lambda z^{\diamond}, p^{\tau}\, \mathbf{cons}_\tau z p \mathbf{nil}_\tau, \mathbf{nil}_\tau\rangle\, \mathbf{t}
\end{aligned}$$

Man erkennt unmittelbar, daß $\lhd_\tau$ in `insert` einmal vorkommt und zwar innerhalb *eines* Paars geschweifter Klammern; außerdem gibt es keine geschachtelten Vorkommen geschweifter Klammern. Es ist also $\vartheta(\texttt{insert}) = X \cdot \vartheta(\lhd_\tau) + O(X)$. Dies entspricht auch dem bekannten Resultat, daß Einfügen in geordnete Listen linear in der Zahl der Vergleichsoperationen ist. Man beachte, wie leicht der vorgestellte Apparat Komplexitätsbetrachtungen bezüglich Unterprogrammaufrufen (also $\lhd_\tau$) zuläßt.

Aus der Funktion `insert` kann leicht ein Sortieralgorithmus definiert werden: $\texttt{insertionsort} = \lambda l^{\tau}\, l\left\{\lambda x^{\diamond}, y^{\tau}, l^{\mathbf{L}(\tau)}\, \texttt{insert}\, lxy\right\} \mathbf{nil}_\tau$. Es ist $\vartheta(\texttt{insertionsort}) = X \cdot \vartheta(\texttt{insert}) + O(X) = X^2 \cdot \vartheta(\lhd_\tau) + O(X^2)$, was wiederum richtig wiedergibt, daß `insertionsort` quadratisch viele Vergleiche benötigt.

Bemerkung 1.4.2. Obige Formulierung von `insertionsort` ist [11] entnommen, wo zunächst ein Rekursor definiert wird, der sich vom (in Gödels T) üblichen dadurch unterscheidet, daß *wahlweise* (im ausschließenden Sinn) auf die Rekursionsvariable oder einen rekursiven Aufruf zurückgegriffen werden kann.

Im Fall von `insert` kommt man jedoch auch einfacher zum Ziel: Man definiert zunächst eine "elementare Sortierfunktion" $\lhd'_\tau$ vom Typ $\tau \multimap \tau \multimap \tau \otimes \tau$, die zwei Werte nimmt und in sortierter Reihenfolge wieder ausgibt.

$$\lhd'_\tau = \lambda p_1^{\tau}, p_2^{\tau} \lhd_\tau p_1 p_2 (\lambda y^{\mathrm{bool}}, p^{\tau\otimes\tau}\, (p(\lambda p_1^{\tau}, p_2^{\tau}\, y\, \langle \otimes_{\tau,\tau} p_1 p_2, \otimes_{\tau,\tau} p_2 p_1\rangle)))$$

Dann kann `insert` *ohne* den Umweg über Paartypen definiert werden:

$$\begin{aligned}
\texttt{insert} = \lambda l\, l &\left\{\lambda x_1^{\diamond}, y_1^{\tau}, p^{\diamond\multimap\tau\multimap\mathbf{L}(\tau)}, x_2^{\diamond}, y_2^{\tau} \lhd'_\tau y_1 y_2 (\lambda z_1^{\tau}, z_2^{\tau}\, \mathbf{cons}_\tau x_1 z_1 (p x_2 z_2))\right\} \\
&(\lambda x^{\diamond}, y^{\tau}\, \mathbf{cons}_\tau x y \mathbf{nil}_\tau)
\end{aligned}$$

Wegen $\vartheta(\texttt{insert}) = X \cdot \vartheta(\lhd'_\tau) + O(X)$ und $\vartheta(\lhd'_\tau) = \vartheta(\lhd_\tau)$ ergeben sich die gleichen polynomialen Schranken wie oben.

[11] Das $\diamond$ im Typ ist nötig, da die Ausgabe ein Konstruktorsymbol mehr enthält als die Eingabe.

2 Vorschlag einer Arithmetik für System $\mathcal{C}$

In diesem Kapitel wird eine linear getypte Fassung ℓHA^ω der Heytingschen Arithmetik vorgestellt. Alle beweisbaren Formeln sind "wahr" in dem Sinne, daß alles Beweisbare auch in der normalen Heytingschen Arithmetik beweisbar ist. Siehe dazu Proposition 2.4.5.

Es wird (in Definition 2.3.1) zu jeder Formel A ein "rechnerischer Gehalt" $[\![A]\!]$ definiert. Dieser gibt (falls die Formel tatsächlich rechnerischen Gehalt hat) die Signatur (also den Typ) eines Programms an, das die Formel realisiert. Ist nun t ein Term dieses Typs, so formalisiert die (in Definition 2.5.1 eingeführte) Schreibweise $t\,\underline{\mathbf{mr}}\,A$ die intuitive Vorstellung, daß t die Formel A "realisiert", also ein Programm zur Berechnung von Beispielen für die in A enthaltenen Existenzquantoren darstellt. So ist etwa für quantorenfreies A die Formel $t\,\underline{\mathbf{mr}}\,\exists^* xA$ gerade $A[t/x]$. Sodann wird (in Definition 2.5.3 und Satz 2.6.2) zu jedem Beweis N ein Term $[\![N]\!]$ in Hofmanns System $\mathcal{C}$ und ein Beweis $\mu(N)$ definiert, so daß $\mu(N)$ ein ℓHA^ω–Beweis von $[\![N]\!]\ \underline{\mathbf{mr}}\,A$ ist. Insbesondere kann also aus jedem Beweis ein polynomialzeitberechenbares Programm "extrahiert" werden.

Diese Methode der "modifizierten Realisierbarkeit" wurde zuerst von Kreisel (für die gewöhnliche Heytingsche Arithmetik) betrachtet. Eine ausführliche Darstellung findet sich in [18, III,§4].

Es bleibt anzumerken, daß ein System mit ähnlichen Eigenschaften (und insbesondere Realisierbarkeit in $\mathcal{C}$) auch dadurch definiert hätte werden können, daß man diejenigen Beweise in der gewöhnlichen Heytingschen Arithmetik als "geeignet" ansieht, bei denen der (im üblichen Sinne) extrahierte Term gerade im gewählten Termsystem liegt. Dieser Ansatz wurde etwa in [17] verfolgt. Obwohl unabhängig vom Termsystem anwendbar, so hat er jedoch den Nachteil, daß auf diese Weise wenig Einblick in das Wesen dieser Beweise gewonnen wird; insbesondere gibt es auf Grund der Definition keine "einfache" Charakterisierung, wann mit geeigneten Beweisen $N^{A\to B}$ und M^A auch $(NM)^B$ geeignet ist (wobei einfach etwa als "ohne expliziten Bezug zur Programmextraktion" zu verstehen ist).

2.1 Formeln

Ausgangspunkt für die Menge der Formeln von ℓHA^ω ist eine als fest gedachte Menge von Prädikatsymbolen fester Stelligkeit und Typisierung. Prädikatsymbole werden durch $P, Q, \ldots$ mitgeteilt. Die Schreibweise $P(\vec{t})$ impliziere, daß die $\vec{t}$ getypt sind und die Typen die für P richtigen sind.

Für jeden Typ τ sei ein zweistelliges Prädikatsymbol $\approx_\tau$ vorhanden. Für $\approx_\tau\ (t,s)$ werde auch $t \approx_\tau s$ geschrieben, beziehungsweise auch $t \approx s$, wenn der Typ klar oder ohne Bedeutung ist.

Definition 2.1.1 (Formeln). Die Menge der Formeln in ℓHA^ω ist induktiv definiert durch

$$A, B ::= \ P(\vec{t}) \mid A \wedge B \mid A \to B \mid \exists^* xA \mid \forall xA \mid \forall^* xA$$

Bei $\exists^* xA$, $\forall xA$ und $\forall^* xA$ wird das x in A gebunden. Der Wirkungsbereich des Quantors ist dabei so eng wie möglich, es sei denn, die Variable wird von einem Punkt gefolgt. In diesem Fall ist der Wirkungsbereich so weit wie möglich. Als Mitteilungszeichen für Formeln werden $A, B, C, \ldots$ verwendet.

Die Konjunktion sei dabei als das "schwache und" zu verstehen. "Schwach" ist das "schwache und" in dem Sinne, daß zwar $A \to B \to (A \wedge B \to C) \to C$ für beliebige A, B und C beweisbar ist, nicht aber $A \wedge B \to (A \to B \to C) \to C$; siehe hierzu die Regeln $(\wedge^+)$ und $(\wedge^-)$ in Definition 2.4.3. Definition 2.5.3 wird zeigen, daß das "schwache und" dem Paar entspricht.

Die (starke) Disjunktion $A \vee^* B$ sei als syntaktische Abkürzung für $\exists^* x^{\mathrm{bool}}.(x \approx \mathbf{t} \to A) \wedge (x \approx \mathbf{f} \to B)$ verstanden. Der Existenzquantor $\exists^*$ ist der gewöhnliche starke Existenzquantor und $\forall$ der gewöhnliche Allquantor. Der "starke Allquantor" $\forall^*$ entspricht der "underlined universal quantification" in [16] bzw. dem $\{\forall\}$ in [4]. Dabei handelt es sich mehr um einen syntaktischen Trick als um einen (bereits) wohlverstanden Quantor: Die starke Allquantifizierung bezeichnet eine für den rechnerischen Gehalt irrelevante Allquantifizierung. Das heißt, ein Beweis von $\forall^* x.A$ darf das x weder für die Allbeseitigung in einer nicht-Harrop-Formel noch als Zeugen für eine Existenzeinführung verwenden. "Stark" ist der starke Allquantor in dem Sinne, daß bei $\forall^* x \exists^* yA$ das y unabhängig von dem gegebenen x ist, daß also dann auch $\exists^* y \forall xA$ gilt und sogar beweisbar ist, wie Korollar 2.6.4 zeigen wird.

2.2 Gleichheit

Die Gleichheit wird in ℓHA^ω in zweifacher Hinsicht behandelt: Zum einen auf der Metaebene dadurch, daß Terme mit gleicher $\mapsto$–Normalform identifiziert werden, wobei $\mapsto$ dahingehend erweitert sein möge, daß die Forderung "ν Numeral" bzw. "ℓ Liste" bei $\mathbf{S}_i d^\diamond \nu \{t_0, t_1\}\, s \mapsto t_i d^\diamond (\nu \{t_0, t_1\}\, s)$ bzw. $\mathbf{cons}_\tau d^\diamond a\ell \{t\}\, s \mapsto t d^\diamond a(\ell \{t\}\, s)$ fallen gelassen wird. Da es stets höchstens ein $\mapsto$-Redukt gibt, ist diese Identifikation entscheidbar.

Außerdem wird noch das Axiomenschema $s \approx t \to A[s/x] \to A[t/x]$, bezeichnet durch die Konstanten $\boxminus_{A,x,s,t}$ als Axiome zum System dazugenommen; ferner noch das Axiomenschema $t \approx t$, bezeichnet durch die Konstanten $\boxminus_t$.

Dies stellt die Minimalforderung an beweisbarer Gleichheitslogik da. Nach Belieben können weitere irrelevante (im Sinne von Definition 2.3.2), "wahre" Gleichheitsaxiome hinzugenommen werden, ohne daß die im folgenden bewiesenen Aussagen falsch werden. Man beachte jedoch, daß die Aussage von Satz 2.6.2 dadurch höchstens schwächer wird, da die Korrektheit der Extraktion dann in einem stärkeren System bewiesen wird.

2.3 Extrahierte Typen und Harropformeln

Zu jeder Formel definieren wir ihren "rechnerischen Gehalt" $[\![A]\!]$. Hierbei handelt es sich um einen Typ, der die Signatur eines möglichen Realisierers angibt bzw. um $\star$, falls die Formel keine rechnerische Aussage beinhaltet.

Definition 2.3.1. Der rechnerische Gehalt $[\![A]\!]$ einer Formel A wird durch Induktion über den Aufbau von A definiert:

$$
\begin{aligned}
[\![P(\vec{t})]\!] &= \star \\
[\![\exists^* x^\tau A]\!] &= \begin{cases} \tau & [\![A]\!] = \star \\ \tau \otimes [\![A]\!] & \text{sonst} \end{cases} \\
[\![\forall x^\tau A]\!] &= \begin{cases} \star & [\![A]\!] = \star \\ \tau \multimap [\![A]\!] & \text{sonst} \end{cases} \\
[\![\forall^* x^\tau A]\!] &= [\![A]\!] \\
[\![A_0 \wedge A_1]\!] &= \begin{cases} [\![A_i]\!] & [\![A_{1-i}]\!] = \star \\ [\![A_0]\!] \times [\![A_1]\!] & \text{sonst} \end{cases} \\
[\![A \to B]\!] &= \begin{cases} [\![B]\!] & [\![A]\!] = \star \\ \star & [\![B]\!] = \star \\ [\![A]\!] \multimap [\![B]\!] & \text{sonst} \end{cases}
\end{aligned}
$$

Definition 2.3.2 (Harropformeln). Eine Formel A heißt Harropformel, falls $[\![A]\!] = \star$. Harropformeln heißen auch *irrelevant*, nicht-Harropformeln *relevant*.

Beispiel 2.3.3 (starke Disjunktion). Seien A und B beide relevant, so ist $[\![A \vee^* B]\!] = \text{bool} \otimes ([\![A]\!] \times [\![B]\!])$. Sind A und B beide irrelevant, so ist $[\![A \vee^* B]\!] = \text{bool}$. Man beachte, daß Disjunktionen *immer* relevant sind.

2.4 Herleitungsterme und Herleitbarkeitsbegriff

Herleitungsterme sollen Herleitungen bezeichnen. Wie beim Termsystem werden zunächst allgemein Terme definiert, anschließend ein Typisierungskalkül angegeben, der die Terme auszeichnet, die für einen korrekten Beweis stehen. Der so bezeichnete Beweis ist dann jedoch eindeutig, wie Lemma 2.4.4 zeigen wird.

Ausgangspunkt für Herleitungsterme sind gewöhnliche Terme (aus $\mathcal{C}$), Axiome (Mitteilungszeichen $\blacksquare$) sowie Annahmevariablen (Mitteilungszeichen $u, v, w, \ldots$). Jede Annahmevariable habe eine ihr zugeordnete Formel als "Typ"[12]. Für jede Formel gebe es unendlich viele Variablen mit dieser Formel aus "Typ". Die Schreibweise u^A bezeichne die Annahmevariable u; sie kann aber nur gebildet werden, wenn u tatsächlich A als "Typ" hat.

Definition 2.4.1 (Herleitungsterme).

$$N, M ::= \blacksquare \mid u^A \mid \lambda u^A N \mid \lambda x^\tau N \mid \lambda^* x^\tau N \mid \langle N, M \rangle \mid NM \mid Nt \mid tN \mid \{N\} \mid \{N, M\}$$

Als Mitteilungszeichen für Herleitungsterme werden $N, M, L, \ldots$ verwendet. Bezüglich gebundener Umbenennung und Vektoren gelten dieselben Konventionen wie für Terme (siehe Seite 7).

[12] "Typ" ist hier im Sinne der Curry-Howard Korrespondenz zu verstehen.

Definition 2.4.2 (verallgemeinerte Kontexte). Ein *verallgemeinerter* Kontext ist eine Menge von (Objekt-)Variablen und Annahmevariablen.

Als Mitteilungszeichen für verallgemeinerte Kontexte werden $\Psi, \Phi, \ldots$ verwendet. Die Schreibweise Ψ, Φ bezeichnet die Vereinigung, kann aber nur gebildet werden, wenn Ψ und Φ disjunkt sind.

Der Begriff eines Beweises wird durch die (unten definierte) Relation $\Psi \,;\, \Phi \vdash N : A$ wiedergegeben. Diese ist zu lesen als "N ist ein Beweis von A mit freien Variablen enthalten in Ψ und essentiell freien Variablen enthalten in Φ". Dabei sollen die "essentiell freien" Variablen gerade die sein, die in der Programmextraktion eine Rolle spielen.

Definition 2.4.3. Die Relation $\Psi \,;\, \Phi \vdash N : A$ wird induktiv definiert durch:

$$\frac{}{\emptyset \,;\, \emptyset \vdash \blacksquare^A : A} \qquad (Axiom)$$

$$\frac{}{u^A \,;\, u^A \vdash u^A : A} \qquad (Annahme)$$

$$\frac{\Psi, u^A \,;\, \Phi, u^A \vdash N : B}{\Psi \,;\, \Phi \vdash \lambda u^A.N : A \to B} \qquad (\to^+)$$

$$\frac{\Psi \,;\, \Phi_1 \vdash N : A \to B \qquad \Psi \,;\, \Phi_2 \vdash M : A}{\Psi \,;\, \Phi_1, \Phi_2 \vdash NM : B} \qquad (\to^-)$$

$$\frac{\Psi, x^\tau \,;\, \Phi, x^\tau \vdash N : A \qquad \texttt{VB!}}{\Psi \,;\, \Phi \vdash \lambda x^\tau.N : \forall x^\tau A} \qquad (\forall^+)$$

$$\frac{\Psi \,;\, \Phi \vdash N : \forall x^\tau A \qquad \Gamma \vdash t^\tau \qquad \Gamma \subset \Psi}{\Psi \,;\, \Phi, \Gamma \vdash Nt : A[t/x]} \qquad (\forall^-)$$

$$\frac{\Psi, x^\tau \,;\, \Phi \vdash N : A \qquad x \notin \Phi \qquad \texttt{VB!}}{\Psi \,;\, \Phi \vdash \lambda^* x^\tau.N : \forall^* x^\tau A} \qquad (\forall^{*+})$$

$$\frac{\Psi \,;\, \Phi \vdash N : \forall^* x^\tau A \qquad \Gamma \vdash t^\tau \qquad \Gamma \subset \Psi}{\Psi \,;\, \Phi \vdash Nt : A[t/x]} \qquad (\forall^{*-})$$

$$\frac{\Psi \,;\, \Phi \vdash N : A \qquad \Psi \,;\, \Phi \vdash M : B}{\Psi \,;\, \Phi \vdash \langle N, M \rangle : A \wedge B} \qquad (\wedge^+)$$

$$\frac{\Psi \,;\, \Phi \vdash N : A \wedge B}{\Psi \,;\, \Phi \vdash N\mathbf{t} : A} \qquad (\wedge_0^-)$$

$$\frac{\Psi \,;\, \Phi \vdash N : A \wedge B}{\Psi \,;\, \Phi \vdash N\mathbf{f} : B} \qquad (\wedge_1^-)$$

Dabei bezeichnet VB! die übliche Variablenbedingung, also x nicht frei in den "Typen" der Annahmevariablen in Ψ.

Bei der nächsten Gruppe von Regeln handelt es sich um $(\exists^{*-})$, $(\otimes - Ind)$, $(\mathrm{bool} - Ind)$, $(\mathbf{L}(\tau) - Ind)$ und $(\mathbf{N} - Ind)$.

$$\frac{\Psi\ ;\ \Phi_1\ \vdash N\ :\ \exists^* x^\tau A \qquad \Psi, x^\tau, u^A\ ;\ \Phi_2, x^\tau, u^A\ \vdash M\ :\ B \qquad \texttt{VB!}}{\Psi\ ;\ \Phi_1, \Phi_2\ \vdash N(\lambda x^\tau, u^A\, M)\ :\ B}$$

$$\frac{\Gamma \vdash t^{\tau\otimes\rho} \qquad \Psi, x_1^\tau, x_2^\rho, u^{t\approx\otimes_{\tau,\rho} x_1^\tau x_2^\rho}\ ;\ \Phi, x_1^\tau, x_2^\rho\ \vdash N\ :\ A[\otimes_{\tau,\rho} x_1 x_2/x] \qquad \Gamma\subset\Psi \qquad \texttt{VB!}}{\Psi\ ;\ \Phi, \Gamma\ \vdash t(\lambda x_1, x_2, u\, N)\ :\ A[t/x]}$$

$$\frac{\Gamma \vdash t^{\mathrm{bool}} \qquad \Psi\ ;\ \Phi\ \vdash N\ :\ t\approx \mathbf{t}\to B \qquad \Psi\ ;\ \Phi\ \vdash M\ :\ t\approx \mathbf{f}\to B \qquad \Gamma\subset\Psi}{\Psi\ ;\ \Phi, \Gamma\ \vdash t\,\langle N, M\rangle\ :\ B}$$

$$\frac{\Gamma \vdash t^{\mathbf{L}(\tau)} \qquad \Psi\ ;\ \emptyset\ \vdash N\ :\ \forall^* x^{\mathbf{L}(\tau)}\forall y^\diamond \forall z^\tau . A \to A[\mathbf{cons}_\tau yzx/x] \qquad \Gamma\subset\Psi}{\Psi\ ;\ \Gamma\ \vdash t\,\{N\}\ :\ A[\mathbf{nil}_\tau/x]\to A[t/x]}$$

$$\frac{\Gamma \vdash t^{\mathbf{N}} \qquad \Psi\ ;\ \emptyset\ \vdash N_i\ :\ \forall^* x^{\mathbf{N}}\forall y^\diamond . A \to A[\mathbf{S}_i yx/x] \qquad \Gamma\subset\Psi}{\Psi\ ;\ \Gamma\ \vdash t\,\{N_1, N_2\}\ :\ A[0/x]\to A[t/x]}$$

Hierbei bezeichne VB! in $(\exists^{*-})$ die Forderung, daß x weder frei in B noch frei in den "Typen" der Annahmevariablen von Ψ vorkommen darf. VB! in $(\otimes - Ind)$ ist die Forderung, daß x_1, x_2 weder frei in den Typen der Annahmevariablen von Ψ noch frei in A vorkommen.

Neben der Regel der Abschwächung, die sich nicht aus den bisherigen Regeln ergibt wird noch eine Regel hinzugenommen, die die Idee der essentiell freien Variablen ausdrückt: essentiell freie Variablen werden nur benötigt um "essentielle Dinge", also relevante Formeln zu beweisen.

$$\frac{\Psi\ ;\ \Phi\ \vdash N\ :\ A \qquad \Psi\subset\Psi' \qquad \Phi\subset\Phi'}{\Psi'\ ;\ \Phi'\ \vdash N\ :\ A} \qquad (\textit{weakening})$$

$$\frac{\Psi\ ;\ \Phi\ \vdash N\ :\ A \qquad [\![A]\!] = \star}{\Psi\ ;\ \emptyset\ \vdash N\ :\ A} \qquad (\textit{pruning})$$

An Axiomen sei außer den Gleichheitsaxiomen noch das Axiom der Existenzeinführung vorhanden: $\boxtimes_{x,A}$ sei ein Axiom vom Typ $\forall x.A \to \exists^* xA$.

Die Schreibweise N^A soll N bezeichnen, aber nur gebildet werden können, wenn es Ψ und Φ gibt, mit $\Psi\ ;\ \Phi\ \vdash N\ :\ A$.

Auch wenn der Herleitbarkeitsbegriff in Sequenzenschreibweise angegeben ist, so enthält der Herleitungsterm alle wesentlichen Informationen, wie das nächste Lemma zeigt:

Lemma 2.4.4. *Für jeden Herleitungsterm N gibt es höchstens eine Formel A, so daß N^A. In diesem Fall gibt es kleinste Ψ und Φ mit $\Psi\ ;\ \Phi\ \vdash N\ :\ A$. Für diese gilt $\Phi\subset\Psi$.*

Beweis. Unmittelbar klar durch Induktion über N. □

In diesem Fall möge N auch Herleitung heißen. Die nach diesem Lemma existenten kleinsten (und dadurch eindeutig bestimmten) Mengen Ψ und Φ heißen auch "Menge der freien Variablen von N" und "Menge der essentiell freien Variablen von N".

Unmittelbar durch Induktion über die Definition von $\Psi\ ;\ \Phi \vdash N : A$ sieht man, daß der Herleitungsbegriff korrekt ist, in dem Sinne, daß sich bei Weglassen des essentiellen Kontexts Φ wieder der übliche Herleitungsbegriff ergibt. Genauer gilt

Proposition 2.4.5 (Korrektheit des Herleitbarkeitsbegriffs). *Ist $\Psi\ ;\ \Phi \vdash N : A$, so ist A nach Ersetzen aller Vorkommen von $\forall^*$ durch $\forall$ auch in der üblichen Heytingschen Arithmetik im Kontext Ψ beweisbar.*

Insbesondere ist die Verwendung von $\forall^*$ damit gerechtfertigt.

2.5 Definition des Realisierbarkeitsbegriffs und der extrahierten Terme

Es wird nun definiert, was es heißt, daß ein Term t eine Formel A realisiert. Informell soll es bedeuten, daß der Term ein "Programm" zu Berechnung der Objekte ist, deren Existenz in A behauptet wird. Formell wird dies durch die *Formel* $t\,\underline{\mathbf{mr}}\,A$ ausgedrückt. Dabei ist klammert $\underline{\mathbf{mr}}$ so eng wie möglich. Da es auch Formeln ohne rechnerischen Gehalt gibt, wird zur Vereinfachung der Schreibweise ein neues Symbol ε als möglicher Realisierer einer Formel A mit $[\![A]\!] = \star$ eingeführt.

Um die Schreibweise zu vereinfachen, werden die Abkürzungen $\pi_0 = \lambda x, y\, x$ und $\pi_1 = \lambda x, y, y$ für die Projektionen verwendet.

Definition 2.5.1 ($\underline{\mathbf{mr}}$). Durch Induktion über A wird eine Formel $t\,\underline{\mathbf{mr}}\,A$ der Arithmetik für beliebiges $t^{[\![A]\!]}$ definiert.

$$
\begin{aligned}
\varepsilon\,\underline{\mathbf{mr}}\,P(\vec{t}) &= P(\vec{t}) \\
r\,\underline{\mathbf{mr}}\,\exists^* A &= \begin{cases} \varepsilon\,\underline{\mathbf{mr}}\,A[r/x] & [\![A]\!] = \star \\ r\pi_1\,\underline{\mathbf{mr}}\,A[r\pi_0/x] & \text{sonst} \end{cases} \\
r\,\underline{\mathbf{mr}}\,\forall x A &= \begin{cases} \forall x.\varepsilon\,\underline{\mathbf{mr}}\,A & [\![A]\!] = \star \\ \forall x.rx\,\underline{\mathbf{mr}}\,A & \text{sonst} \end{cases} \\
r\,\underline{\mathbf{mr}}\,\forall^* x A &= \forall x.r\,\underline{\mathbf{mr}}\,A \\
r\,\underline{\mathbf{mr}}\,(A \to B) &= \begin{cases} \varepsilon\,\underline{\mathbf{mr}}\,A \to r\,\underline{\mathbf{mr}}\,B & [\![A]\!] = \star \\ \forall x.x\,\underline{\mathbf{mr}}\,A \to \varepsilon\,\underline{\mathbf{mr}}\,B & [\![A]\!] \neq \star = [\![B]\!] \\ \forall x.x\,\underline{\mathbf{mr}}\,A \to rx\,\underline{\mathbf{mr}}\,B & \text{sonst} \end{cases} \\
r\,\underline{\mathbf{mr}}\,(A \wedge B) &= \begin{cases} \varepsilon\,\underline{\mathbf{mr}}\,A \wedge r\,\underline{\mathbf{mr}}\,B & [\![A]\!] = \star \\ r\,\underline{\mathbf{mr}}\,A \wedge \varepsilon\,\underline{\mathbf{mr}}\,B & [\![B]\!] = \star \\ r\mathbf{t}\,\underline{\mathbf{mr}}\,A \wedge r\mathbf{f}\,\underline{\mathbf{mr}}\,B & \text{sonst} \end{cases}
\end{aligned}
$$

Aus der Definition sieht man (durch Induktion über A) unmittelbar, daß $r\,\underline{\mathbf{mr}}\,A$ kein $\exists^*$ enthält. Insbesondere ist also

Proposition 2.5.2 (Irrevelanz der Realisierbarkeitsformel). $\llbracket r\,\underline{\mathbf{mr}}\,A\rrbracket = \star$

Es sei x_{u^A} im Falle von $\llbracket A\rrbracket \neq \star$ eine zu u^A eindeutig bestimmte und von den "bisherigen Variablen" hinreichend disjunkte Variable vom Typ $\llbracket A\rrbracket$. Im Falle von $\llbracket A\rrbracket = \star$ ist $x_{u^A} = \varepsilon$. Ferner ist $\widetilde{u^A}$ eine hinreichend neue Annahmevariable für die Formel $x_{u^A}\,\underline{\mathbf{mr}}\,A$.

Definition 2.5.3 (extrahierte Terme). Der extrahierte Term $\llbracket N\rrbracket$ einer Herleitung N ist induktiv definiert:

$$
\begin{aligned}
\llbracket \boxtimes_{x^\tau, A}\rrbracket &= \begin{cases} (\lambda x^\tau\, x) & \llbracket A\rrbracket = \star \\ \otimes_{\tau, \llbracket A\rrbracket} & \text{sonst} \end{cases} \\
\llbracket \boxminus_{A,x,s,t}\rrbracket &= \begin{cases} \varepsilon & \llbracket A\rrbracket = \star \\ \lambda x^{\llbracket A\rrbracket}\, x & \text{sonst} \end{cases} \\
\llbracket \boxminus_t\rrbracket &= \varepsilon \\
\llbracket u^A\rrbracket &= x_{u^A} \\
\llbracket \lambda u^A\, N^B\rrbracket &= \begin{cases} \varepsilon & \llbracket B\rrbracket = \star \\ \llbracket N\rrbracket & \llbracket A\rrbracket = \star \neq \llbracket B\rrbracket \\ \lambda x_u\ \llbracket N\rrbracket & \text{sonst} \end{cases} \\
\llbracket N^{A\to B} M\rrbracket &= \begin{cases} \varepsilon & \llbracket B\rrbracket = \star \\ \llbracket N\rrbracket & \llbracket A\rrbracket = \star \neq \llbracket B\rrbracket \\ \llbracket N\rrbracket\ \llbracket M\rrbracket & \text{sonst} \end{cases} \\
\llbracket \lambda x^\tau\, N^A\rrbracket &= \begin{cases} \varepsilon & \llbracket A\rrbracket = \star \\ \lambda x\ \llbracket N\rrbracket & \text{sonst} \end{cases} \\
\llbracket N^{\forall x.A} t\rrbracket &= \begin{cases} \varepsilon & \llbracket A\rrbracket = \star \\ \llbracket N\rrbracket\ t & \text{sonst} \end{cases} \\
\llbracket \lambda^* x^\tau N\rrbracket &= \llbracket N\rrbracket \\
\llbracket N^{\forall^* x.A} t\rrbracket &= \llbracket N\rrbracket \\
\llbracket \langle N, M\rangle\rrbracket &= \begin{cases} \varepsilon & \llbracket A\rrbracket = \star = \llbracket B\rrbracket \\ \llbracket N\rrbracket & \llbracket A\rrbracket \neq \star = \llbracket B\rrbracket \\ \llbracket M\rrbracket & \llbracket A\rrbracket = \star \neq \llbracket B\rrbracket \\ \langle \llbracket N\rrbracket, \llbracket M\rrbracket\rangle & \text{sonst} \end{cases} \\
\llbracket N^{A\wedge B}\mathbf{t}\rrbracket &= \begin{cases} \varepsilon & \llbracket A\rrbracket = \star \\ \llbracket N\rrbracket & \llbracket B\rrbracket = \star \neq \llbracket A\rrbracket \\ \llbracket N\rrbracket\,\mathbf{t} & \text{sonst} \end{cases} \\
\llbracket N^{A\wedge B}\mathbf{f}\rrbracket &= \begin{cases} \varepsilon & \llbracket B\rrbracket = \star \\ \llbracket N\rrbracket & \llbracket A\rrbracket = \star \neq \llbracket B\rrbracket \\ \llbracket N\rrbracket\,\mathbf{f} & \text{sonst} \end{cases}
\end{aligned}
$$

$$\llbracket N^{\exists^* x A}(\lambda x, u\, M^B)\rrbracket = \begin{cases} \varepsilon & \llbracket B\rrbracket = \star \\ (\lambda x\ \llbracket M\rrbracket)\,\llbracket N\rrbracket & \llbracket A\rrbracket = \star \neq \llbracket B\rrbracket \\ \llbracket N\rrbracket\,(\lambda x, x_u\ \llbracket M\rrbracket) & \text{sonst} \end{cases}$$

$$\llbracket t^{\tau\otimes\rho}(\lambda x_1, x_2, u\, N^A)\rrbracket = \begin{cases} \varepsilon & \llbracket A\rrbracket = \star \\ t(\lambda x_1, x_2\ \llbracket N\rrbracket) & \text{sonst} \end{cases}$$

$$\llbracket t^{\mathrm{bool}}\left\langle N^{t\approx\mathbf{t}\to A}, M\right\rangle\rrbracket = \begin{cases} \varepsilon & \llbracket A\rrbracket = \star \\ t\,\langle\llbracket N\rrbracket, \llbracket M\rrbracket\rangle & \text{sonst} \end{cases}$$

$$\llbracket t^{\mathbf{L}(\tau)}\left\{N^B\right\}\rrbracket = \begin{cases} \varepsilon & \llbracket B\rrbracket = \star \\ t\,\{N\} & \text{sonst} \end{cases}$$

$$\llbracket t^{\mathbf{N}}\left\{N_0^B, N_1\right\}\rrbracket = \begin{cases} \varepsilon & \llbracket B\rrbracket = \star \\ t\,\{\llbracket N_0\rrbracket, \llbracket N_1\rrbracket\} & \text{sonst} \end{cases}$$

Bemerkung 2.5.4. Aus der Definition von $\llbracket N\rrbracket$ durch Induktion über den Aufbau von N ergibt sich unmittelbar, daß zu gegebenem N der Term $\llbracket N\rrbracket$ in polynomialer Zeit gefunden werden kann. Es ist also nicht nur das "extrahierte Programm" in `Ptime`, sondern auch der Vorgang der Programmextraktion selbst.

Für einen verallgemeinerten Kontext Ψ entstehe $\overline{\Psi}$ aus Ψ, indem jedes u^A durch die beiden Variablen x_{u^A} und $\widetilde{u^A}$ ersetzt wird bzw. nur durch $\widetilde{u^A}$, falls $\llbracket A\rrbracket = \star$ (Im allgemeinen hat $\overline{\Psi}$ also mehr Elemente als Ψ.). Außerdem sei $\llbracket\vec{N}\rrbracket$ kurz für $\llbracket N_1\rrbracket, \dots, \llbracket N_n\rrbracket$.

2.6 Korrektheit der Extraktion

Zur Vereinfachung der Schreibweise wird die Relation $\cdot \vdash \cdot$ dahingehend erweitert, daß $\Gamma \vdash \varepsilon^\star$ für beliebige Γ gelten möge. Außerdem seien ε, Γ und Γ, ε als Schreibweise für Γ zu lesen.

Im nächsten Satz mag es zunächst verwirrend aussehen, daß extrahierte Terme in der (Ober-)menge der freien Variablen vorkommen. Dieser scheinbare Widerspruch löst sich jedoch, wenn man sich erinnert, daß $\llbracket\vec{v}\rrbracket = x_{v_1}, \dots, x_{v_n}$ ist.

Satz 2.6.1 (Soundness of typing). $\vec{u}, \vec{x}\ ;\ \vec{v}, \vec{y} \vdash N : A \Longrightarrow \llbracket\vec{v}\rrbracket, \vec{y} \vdash \llbracket N\rrbracket^{\llbracket A\rrbracket}$

Beweis. Der Fall $\llbracket A\rrbracket = \star$ ist wegen $\Gamma \vdash \varepsilon^\star$ trivial. Betrachte also nur die Fälle, in denen die bewiesene Formel relevant ist. Bei der folgenden Induktion seien alle Bezeichnungen wie in der Definition von $\cdot\,;\,\cdot \vdash \cdot : \cdot$ gewählt (vgl. Seite 23).

Induktion($\vec{u}, \vec{x}\ ;\ \vec{v}, \vec{y} \vdash N : A$)

$\boxtimes_{x^\tau, A}$ **Fall** $\llbracket A\rrbracket = \star$ Es ist $\emptyset \vdash (\lambda x^\tau\ x)^{\tau\multimap\tau}$.

Fall sonst Es ist $\emptyset \vdash \otimes_{\tau, \llbracket A\rrbracket}{}^{\tau\multimap\llbracket A\rrbracket\multimap\tau\otimes\llbracket A\rrbracket}$.

$\boxminus_{A,x,s,t}$ Im (einzig interessanten) Fall $\llbracket A\rrbracket \neq \star$ ist $\llbracket s \approx t \to A[s/x] \to A[t/x]\rrbracket = \llbracket A\rrbracket \multimap \llbracket A\rrbracket$, was auch der Typ von $\mu\,(\boxminus_{A,x,s,t})$ ist.

$\boxminus_t$ Es ist *stets* $[\![t \approx t]\!] = \star$.

u^A Es ist $x_u \vdash x_u{}^{[\![A]\!]}$.

$(\to^+)$ Sei $\Phi = \vec{u}, \vec{x}$.

Fall $[\![A]\!] = \star$ Dann ist wegen $x_u = \varepsilon$ nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{x} \vdash [\![N]\!]^{[\![B]\!]}$, was auch zu zeigen ist.

Fall sonst Dann ist nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{x}, x_u \vdash [\![N]\!]^{[\![B]\!]}$. Also $[\![\vec{u}]\!], \vec{x}, \vdash (\lambda x_u . [\![N]\!])^{[\![A]\!] \multimap [\![B]\!]}$ nach $(\multimap^+)$.

$(\to^-)$ Sei $\Phi_1 = \vec{u}, \vec{x}$ und $\Phi_2 = \vec{v}, \vec{y}$.

Fall $[\![A]\!] = \star$ Dann ist nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{x} \vdash [\![N]\!]^{[\![B]\!]}$, was auch die Behauptung ist.

Fall sonst Dann ist nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{x} \vdash [\![N]\!]^{[\![A]\!] \multimap [\![B]\!]}$ und $[\![\vec{v}]\!], \vec{y} \vdash [\![M]\!]^{[\![A]\!]}$, so daß mit $(\multimap^-)$ die Behauptung folgt.

$(\forall^+)$ Sei $\Phi = \vec{u}, \vec{x}$. Dann ist nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{x}, x^\tau \vdash [\![N]\!]^{[\![A]\!]}$, also $[\![\vec{u}]\!], \vec{x} \vdash (\lambda x^\tau\ [\![N]\!])^{\tau \multimap [\![A]\!]}$ nach $(\multimap^+)$.

$(\forall^-)$ Sei $\Phi = \vec{u}, \vec{x}$. Dann ist nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{x} \vdash [\![N]\!]^{\tau \multimap [\![A]\!]}$. Wegen $\Gamma \vdash t^\tau$ ist $[\![\vec{u}]\!], \vec{x}, \Gamma \vdash ([\![N]\!]\ t)^{[\![A]\!]}$ nach $(\multimap^-)$.

$(\forall^{*+})$ Hier fallen Induktionsvoraussetzung und Behauptung zusammen.

$(\forall^{*-})$ Hier fallen Induktionsvoraussetzung und Behauptung zusammen.

$(\wedge^{+/-})$ Ergibt sich jeweils unmittelbar aus der Induktionsvoraussetzung.

$(\exists^{*-})$ Sei $\Phi_1 = \vec{u}, \vec{x}$ und $\Phi_2 = \vec{v}, \vec{y}$.

Fall $[\![A]\!] = \star$ Dann ist nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{x} \vdash [\![N]\!]^\tau$ und $[\![\vec{v}]\!], \vec{y}, x^\tau \vdash [\![M]\!]^{[\![B]\!]}$. Also $[\![\vec{v}]\!], \vec{y}, [\![\vec{u}]\!], \vec{x} \vdash (\lambda x\ [\![M]\!])\ [\![N]\!]^{[\![B]\!]}$ nach $(\multimap^+)$ und $(\multimap^-)$.

Fall sonst Dann ist nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{x} \vdash [\![N]\!]^{\tau \otimes [\![A]\!]}$ und $[\![\vec{v}]\!], \vec{y}, x^\tau, x_u \vdash [\![M]\!]^{[\![B]\!]}$. Mit $(\otimes^-)$ ist dann $[\![\vec{u}]\!], \vec{x}, [\![\vec{v}]\!], \vec{y} \vdash [\![N]\!]\ (\lambda x, x_u\ [\![M]\!])^{[\![B]\!]}$.

$(\otimes - Ind)$ Sei $\Phi = \vec{u}, \vec{y}$. Dann ist nach Induktionsvoraussetzung $[\![\vec{u}]\!], \vec{y}, x_1^\tau, x_2^\tau \vdash [\![N]\!]^{[\![A]\!]}$, also wegen $(\otimes^-)$ auch $\Gamma, [\![\vec{u}]\!], \vec{y} \vdash t(\lambda x_1, x_2\ [\![N]\!])^{[\![A]\!]}$

$(\text{bool} - Ind)$ Sei $\Phi = \vec{u}, \vec{x}$. Nach Induktionsvoraussetzung ist $[\![\vec{u}]\!], \vec{x} \vdash [\![N]\!]^{[\![B]\!]}$ und $[\![\vec{u}]\!], \vec{x} \vdash [\![M]\!]^{[\![B]\!]}$. Wegen (bool^-) ist $\Gamma, [\![\vec{u}]\!], \vec{x} \vdash (t \langle [\![N]\!], [\![M]\!] \rangle)^{[\![B]\!]}$.

$(\mathbf{L}(\tau) - Ind)$ Nach Induktionsvoraussetzung ist $[\![N]\!]$ ein geschlossener Term vom Typ $\diamond \multimap \tau \multimap [\![A]\!] \multimap [\![A]\!]$. Wegen $(\mathbf{L}(\tau)^-)$ ist also $\Gamma \vdash t\,\{[\![N]\!]\}^{[\![A]\!]}$.

($\mathbf{N}-Ind$) Nach Induktionsvoraussetzung sind $[\![N_0]\!]$ und $[\![N_1]\!]$ geschlossene Terme vom Typ $\diamond \multimap [\![A]\!] \multimap [\![A]\!]$. Wegen $\Gamma \vdash t^{\mathbf{N}}$ ist also $\Gamma \vdash t\,\{\,[\![N_0]\!]\,,\,[\![N_1]\!]\,\}^{[\![A]\!]\multimap[\![A]\!]}$. Beachte, daß $\Gamma = \{\vec{y}\}$.

($pruing$) Klar, da sogar $\emptyset \vdash \varepsilon^{\star}$.

($weakening$) trivial (wegen Proposition 1.1.6).

□

Der folgende Satz zeigt die Korrektheit der Programmextraktion. Wegen Proposition 2.4.5 ist die Aussage, daß $[\![N]\!]\ \underline{\mathbf{mr}}\, A$ beweisbar ist *stärker* als die Aussage, daß $[\![N]\!]\ \underline{\mathbf{mr}}\, A$ "wahr" ist.

Satz 2.6.2 (Soundness). *Sei* $\Psi\ ;\ \Phi\ \vdash N\ :\ A$ *mit* $\Phi \subset \Psi$. *Dann gibt es* $\mu\,(N)$ *mit* $\overline{\Psi}\ ;\ \emptyset\ \vdash \mu\,(N)\ :\ [\![N]\!]\ \underline{\mathbf{mr}}\, A$.

Beweis. Induktion($\Psi\ ;\ \Phi\ \vdash N\ :\ A$) Die Bezeichnungen seinen wie in der Definition von der Relation $\cdot\ ;\ \cdot\ \vdash \cdot\ :\ \cdot$ gewählt (vgl. Seite 23).

Wegen Proposition 2.5.2 genügt es, $\overline{\Psi}\ ;\ \Phi'\ \vdash \mu\,(N)\ :\ [\![N]\!]\ \underline{\mathbf{mr}}\, A$ mit beliebigem Φ' zu zeigen. Eine anschließende Anwendung der Regel ($pruning$) liefert dann das Gewünschte. Außerdem wird im Beweis stillschweigend die Erkenntnis benutzt, daß *alle* auftretenden Formeln irrelevant sind und also an beliebiger Stelle durch ($pruning$) essentiell freie Variable beseitigt werden können. Außerdem fallen bei ($pruning$) und ($weaking$) offenbar Induktionsvoraussetzung und Behauptung zusammen. Außerdem fallen bei $\boxminus_t$ Voraussetzung und Behauptung zusammen.

$\boxtimes_{x^\tau,A}$ $\underline{\textbf{Fall}\ [\![A]\!] = \star}$ Es ist zu zeigen, daß die Formel $(\lambda x^\tau.x)\,\underline{\mathbf{mr}}\,\forall x^\tau.A \to \exists^* x^\tau A$ geschlossen herleitbar ist. Es ist

$$\begin{aligned} & (\lambda x^\tau.x)\ \underline{\mathbf{mr}}\,\forall x^\tau.A \to \exists^* x^\tau A \\ = \;& \forall x^\tau.\,\underbrace{(\lambda x^\tau.x)x}_{x}\ \underline{\mathbf{mr}}\,(A \to \exists^* x^\tau A) \\ = \;& \forall x^\tau.\varepsilon\ \underline{\mathbf{mr}}\,A \to x\ \underline{\mathbf{mr}}\,\exists^* x^\tau A \\ = \;& \forall x^\tau.\varepsilon\ \underline{\mathbf{mr}}\,A \to x\ \underline{\mathbf{mr}}\,\exists^* x^\tau A \\ = \;& \forall x^\tau.\varepsilon\ \underline{\mathbf{mr}}\,A \to \varepsilon\ \underline{\mathbf{mr}}\,A \end{aligned}$$

Also kann $\mu\,(N) = \lambda x, u^{\varepsilon\,\underline{\mathbf{mr}}\,A}\ u$ gesetzt werden.

$\underline{\textbf{Fall}\ \text{sonst}}$ Zu zeigen ist, daß die Formel $\otimes_{\tau,[\![A]\!]}\ \underline{\mathbf{mr}}\,\forall x^\tau.A \to \exists^* x^\tau A$ geschlossen herleitbar ist. Es ist

$$\begin{aligned} & \otimes_{\tau,[\![A]\!]}\ \underline{\mathbf{mr}}\,\forall x^\tau.A \to \exists^* x^\tau A \\ = \;& \forall x^\tau.\otimes_{\tau,[\![A]\!]}\,x\ \underline{\mathbf{mr}}\,(A \to \exists^* x^\tau A) \\ = \;& \forall x^\tau y^{[\![A]\!]}.y\ \underline{\mathbf{mr}}\,A \to \otimes_{\tau,[\![A]\!]}\,xy\ \underline{\mathbf{mr}}\,\exists^* x^\tau A \\ = \;& \forall x^\tau y^{[\![A]\!]}.y\ \underline{\mathbf{mr}}\,A \to \underbrace{\otimes_{\tau,[\![A]\!]}\,xy\pi_1}_{y}\ \underline{\mathbf{mr}}\,A[\underbrace{\otimes_{\tau,[\![A]\!]}\,xy\pi_0}_{x}/x] \end{aligned}$$

Es kann also $\mu\,(\boxtimes_{x^\tau,A}) = \lambda x^\tau, y^{[\![A]\!]}, u^{y\,\underline{\mathbf{mr}}\,A}\ u$ gesetzt werden.

$\boxminus_{A,x,s,t}$ $\underline{\textbf{Fall } [\![A]\!] = \star}$ Es ist $s \approx t \to \varepsilon\, \underline{\mathbf{mr}}\, A[s/x] \to \varepsilon\, \underline{\mathbf{mr}}\, A[t/x]$ wieder ein Gleichheitsaxiom. Also kann $\mu\left(\boxminus_{A,x,s,t}\right) = \boxminus_{\varepsilon\, \underline{\mathbf{mr}}\, A,x,s,t}$ gesetzt werden.

$\underline{\textbf{Fall} \text{ sonst}}$ Es ist zu zeigen:

$$s \approx t \to \forall y^{[\![A]\!]}.y\, \underline{\mathbf{mr}}\, A[s/x] \to \underbrace{(\lambda x^{[\![A]\!]}.x)y}_{y}\, \underline{\mathbf{mr}}\, A[t/x]$$

Ein Beweis hierfür ist $\lambda u^{s\approx t} \lambda y\ \boxminus_{y\, \underline{\mathbf{mr}}\, A,x,s,t}\ u$.

u^A In der Tat $\widetilde{u}^{x_u\, \underline{\mathbf{mr}}\, A}\ ;\ \emptyset\ \vdash \widetilde{u}\ :\ x_u\, \underline{\mathbf{mr}}\, A$.

$(\to^+)$ $\underline{\textbf{Fall } [\![A]\!] = \star}$ Nach Induktionsvoraussetzung $\overline{\Psi}, \widetilde{u}^{\varepsilon\, \underline{\mathbf{mr}}\, A}\ ;\ \emptyset\ \vdash \mu(N)\ :\ [\![N]\!]\ \underline{\mathbf{mr}}\, B$. Daraus ergibt sich mit $(\to^+)$, daß $\lambda\widetilde{u}\,\mu(N)$ ein Beweis von $\varepsilon\, \underline{\mathbf{mr}}\,(A \to B)$ aus den richtigen Annahmen ist.

$\underline{\textbf{Fall } [\![A]\!] \neq \star = [\![A]\!]}$ Nach Induktionsvoraussetzung $\overline{\Psi}, x_u, \widetilde{u}\ ;\ \emptyset\ \vdash \mu(N)\ :\ \varepsilon\, \underline{\mathbf{mr}}\, B$. Daraus ergibt sich mit $(\to^+)$ und $(\forall^+)$ die Behauptung.

$\underline{\textbf{Fall} \text{ sonst}}$ Nach Induktionsvoraussetzung $\overline{\Psi}, x_u, \widetilde{u}\ ;\ \emptyset\ \vdash \mu(N)\ :\ [\![N]\!]\ \underline{\mathbf{mr}}\, B$. Daraus ergibt sich

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \lambda x_u \lambda\widetilde{u}.\mu(N)\ :\ \forall x_u.x_u\, \underline{\mathbf{mr}}\, A \to \underbrace{[\![N]\!]}_{(\lambda x_u.[\![N]\!])x_u}\ \underline{\mathbf{mr}}\, B$$

via $(\to^+\ \widetilde{u})$ und $(\forall^+ x_u)$.

$(\to^-)$ $\underline{\textbf{Fall } [\![A]\!] = \star}$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu(N)\ :\ \varepsilon\, \underline{\mathbf{mr}}\, A \to [\![N]\!]\ \underline{\mathbf{mr}}\, B$$

und

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu(M)\ :\ \varepsilon\, \underline{\mathbf{mr}}\, A$$

Also leistet $\mu(N)\,\mu(M)$ das Gewünschte.

$\underline{\textbf{Fall } [\![A]\!] \neq \star = [\![B]\!]}$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu(N)\ :\ \forall x^{[\![A]\!]}.x\, \underline{\mathbf{mr}}\, A \to \varepsilon\, \underline{\mathbf{mr}}\, B$$

und

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu(M)\ :\ [\![M]\!]\ \underline{\mathbf{mr}}\, A$$

Also ist $\mu(N)\ [\![M]\!]\ \mu(M)$ eine Herleitung von $\varepsilon\, \underline{\mathbf{mr}}\, B$.

$\underline{\textbf{Fall} \text{ sonst}}$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu(N)\ :\ \forall x^{[\![A]\!]}.x\, \underline{\mathbf{mr}}\, A \to [\![N]\!]\ x\, \underline{\mathbf{mr}}\, B$$

und

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu(M)\ :\ [\![M]\!]\ \underline{\mathbf{mr}}\, A$$

Also ist $\mu(N)\ [\![M]\!]\ \mu(M)$ eine Herleitung von $[\![N]\!]\ [\![M]\!]\ \underline{\mathbf{mr}}\, B$.

($\forall^+$) $\underline{\textbf{Fall } [\![A]\!] = \star}$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi}, x^\tau \ ; \ \emptyset \ \vdash \mu(N) \ : \ \varepsilon \, \underline{\mathbf{mr}} \, A$$

Daraus ergibt sich mit ($\forall^+$) die Behauptung.

$\underline{\textbf{Fall} \text{ sonst}}$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi}, x^\tau \ ; \ \emptyset \ \vdash \mu(N) \ : \ [\![N]\!] \ \underline{\mathbf{mr}} \, A$$

Daraus ergibt sich mit ($\forall^+$)

$$\overline{\Psi} \ ; \ \emptyset \ \vdash \lambda x^\tau . \mu(N) \ : \ \forall x^\tau . \ \underbrace{[\![N]\!]}_{(\lambda x . [\![N]\!])x} \ \underline{\mathbf{mr}} \, A$$

($\forall^-$) $\underline{\textbf{Fall } [\![A]\!]}$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi} \ ; \ \emptyset \ \vdash \mu(N) \ : \ \forall x^\tau . \varepsilon \, \underline{\mathbf{mr}} \, A$$

Also ist $\mu(N)\, t$ eine geeignete Herleitung.

$\underline{\textbf{Fall} \text{ sonst}}$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi} \ ; \ \emptyset \ \vdash \mu(N) \ : \ \forall x^\tau . \ [\![N]\!] \ x \, \underline{\mathbf{mr}} \, A$$

Also ist $\mu(N)\, t$ eine Herleitung von $[\![N]\!] \ t \, \underline{\mathbf{mr}} \, A[t/x]$.

($\forall^{*+}$) Nach Induktionsvoraussetzung ist

$$\overline{\Psi}, x^\tau \ ; \ \emptyset \ \vdash \mu(N) \ : \ [\![N]\!] \ \underline{\mathbf{mr}} \, A$$

Daraus ergibt sich mit ($\forall^+$)

$$\overline{\Psi} \ ; \ \emptyset \ \vdash \lambda x^\tau . \mu(N) \ : \ \forall x^\tau . \ [\![N]\!] \ \underline{\mathbf{mr}} \, A$$

($\forall^{*-}$) Nach Induktionsvoraussetzung ist

$$\overline{\Psi} \ ; \ \emptyset \ \vdash \mu(N) \ : \ \forall x^\tau . \ [\![N]\!] \ \underline{\mathbf{mr}} \, A$$

Also ist $\mu(N)\, t$ eine Herleitung von $[\![N]\!] \ \underline{\mathbf{mr}} \, A[t/x]$. Beachte hierzu, daß $x \notin \mathrm{FV}([\![N]\!])$.

($\wedge^{+/-}$) Ergibt sich unmittelbar aus der Induktionsvoraussetzung.

($\exists^{*-}$) $\underline{\textbf{Fall } [\![A]\!] = \star}$ Nach Induktionsvoraussetzung gibt es Herleitungen $\mu(N)$ und $\mu(M)$ mit $\overline{\Psi} \ ; \ \emptyset \ \vdash \mu(N) \ : \ \varepsilon \, \underline{\mathbf{mr}} \, A[[\![N]\!]/x]$ und $\overline{\Psi}, x^\tau, \widetilde{u^A}^{\varepsilon \, \underline{\mathbf{mr}} \, A} \ ; \ \emptyset \ \vdash \mu(M) \ :$ $[\![M]\!] \ \underline{\mathbf{mr}} \, B$. Also ist wegen $x \notin \mathrm{FV}(B)$

$$\overline{\Psi} \ ; \ \emptyset \ \vdash (\lambda x \widetilde{u} . \mu(M)) \ [\![N]\!] \ \mu(N) \ : \ \underbrace{[\![M]\!] \ [[\![N]\!]/x]}_{(\lambda x . [\![M]\!])[\![N]\!]} \ \underline{\mathbf{mr}} \, B$$

Fall sonst Nach Induktionsvoraussetzung gibt es Herleitungen $\mu(N)$ und $\mu(M)$ mit $\overline{\Psi}\ ;\ \emptyset\ \vdash \mu(N)\ :\ [\![N]\!]\,\pi_1\,\underline{\mathbf{mr}}\,A[[\![N]\!]\,\pi_0/x]$ und $\overline{\Psi}, x^\tau, x_{u^A}, \widetilde{u^A}\ ;\ \emptyset\ \vdash \mu(M)\ :\ [\![M]\!]\ \underline{\mathbf{mr}}\,B$.

Aus $\mu(M)$, Gleichheitslogik und der Annahme v vom Typ $[\![N]\!] \approx \otimes_{\tau,[\![A]\!]}\,x_1x_2$ erhält man (mit $\mu(N)' = \boxminus_{x'\pi_1\,\underline{\mathbf{mr}}\,A[x'\pi_0/x],x',[\![N]\!],\otimes_{\tau,[\![A]\!]}x_1x_2}v\mu(N)$)

$$\overline{\Psi}, v, x_1, x_2\ ;\ \emptyset\ \vdash \mu(N)'\ :\ x_2\,\underline{\mathbf{mr}}\,A[x_1/x]$$

Wegen der bei $(\exists^{*-})$ geltenden Variablenbedingung erhält man aus $\mu(M)$ durch Substitution

$$\overline{\Psi}, x_1, x_2, \widetilde{u}'\ ;\ \emptyset\ \vdash \mu(M)\,[x_2, x_1/x_u, x]\ :\ [\![M]\!]\,[x_2, x_1/x_u.x]\,\underline{\mathbf{mr}}\,B$$

Weiter durch $(\rightarrow^+)$ ergibt sich

$$\begin{aligned}\overline{\Psi}, x_1, x_2\ ;\ \emptyset\ \vdash \lambda\widetilde{u}'.\mu(M)\,[x_2, x_1/x_u, x]\ :\ & x_2\,\underline{\mathbf{mr}}\,A[x_1/x]\\ & \rightarrow [\![M]\!]\,[x_2, x_1/x_u.x]\,\underline{\mathbf{mr}}\,B\end{aligned}$$

Mit $(\rightarrow^-)$ und $\mu(N)'$ erhält man daraus schließlich

$$\overline{\Psi}, v, x_1, x_2\ ;\ \emptyset\ \vdash (\lambda\widetilde{u}.\mu(M)\,[x_2, x_1/x_u, x])\mu(N)'\ :\ \underbrace{[\![M]\!]\,[x_2, x_1/x_u.x]}_{\otimes_{\tau,[\![A]\!]}x_1x_2(\lambda x x_u.[\![N]\!])}$$

$(\otimes - Ind)$ für $[\![N]\!]$ liefert daraus die Behauptung.

$(\otimes - Ind)$ **Fall** $[\![A]\!] = \star$ Nach Induktionsvoraussetzung ist wegen $x_u = \varepsilon$

$$\overline{\Psi}, x_1^\tau, x_2^\rho, \widetilde{u}^{\varepsilon\,\underline{\mathbf{mr}}\,t\approx\otimes_{\tau,\rho}x_1x_2}\ ;\ \emptyset\ \vdash \mu(N)\ :\ \varepsilon\,\underline{\mathbf{mr}}\,A[\otimes_{\tau,\rho}x_1x_2/x]$$

$(\otimes - Ind\quad t)$ liefert also

$$\overline{\Psi}\ ;\ \emptyset\ \vdash t(\lambda x_1x_2.\mu(N))\ :\ \varepsilon\,\underline{\mathbf{mr}}\,A[t/x]$$

Fall sonst Nach Induktionsvoraussetzung ist wegen $x_u = \varepsilon$

$$\overline{\Psi}, x_1^\tau, x_2^\rho, \widetilde{u}^{\varepsilon\,\underline{\mathbf{mr}}\,t\approx\otimes_{\tau,\rho}x_1x_2}\ ;\ \emptyset\ \vdash \mu(N)\ :\ [\![N]\!]\ \underline{\mathbf{mr}}\,A[\otimes_{\tau,\rho}x_1x_2/x]$$

$(\otimes - Ind\quad t)$ liefert also wegen $[\![N]\!] = \otimes_{\tau,\rho}x_1x_2(\lambda x_1x_2.\,[\![N]\!])$

$$\overline{\Psi}\ ;\ \emptyset\ \vdash t(\lambda x_1x_2.\mu(N))\ :\ t(\lambda x_1x_2.\,[\![N]\!])\,\underline{\mathbf{mr}}\,A[t/x]$$

$(\mathrm{bool} - Ind)$ **Fall** $[\![A]\!] = \star$ wie der Fall "sonst", jedoch alle vorkommenden realisierenden Terme als ε gelesen.

Fall sonst Nach Induktionsvoraussetzung ist

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu(N)\ :\ t \approx \mathbf{t} \rightarrow [\![N]\!]\ \underline{\mathbf{mr}}\,B$$

und

$$\overline{\Psi}\,;\,\emptyset\,\vdash \mu\,(M)\,:\,t\approx\mathbf{f}\to [\![M]\!]\ \underline{\mathbf{mr}}\,B$$

Sei u eine Annahmevariable des Typs $t \approx \mathbf{t}$ und v eine Annahmevariable des Typs $t \approx \mathbf{f}$. Dann

$$\overline{\Psi}, u\,;\,\emptyset\,\vdash \mu\,(N)\,u\,:\quad \underbrace{[\![N]\!]}_{\mathbf{t}\langle\,[\![N]\!]\,,[\![M]\!]\,\rangle}\quad \underline{\mathbf{mr}}\,B$$

und

$$\overline{\Psi}, v\,;\,\emptyset\,\vdash \mu\,(M)\,v\,:\quad \underbrace{[\![M]\!]}_{\mathbf{f}\langle\,[\![N]\!]\,,[\![M]\!]\,\rangle}\quad \underline{\mathbf{mr}}\,B$$

Mit Gleichheitslogik und anschließendem $(\to^+)$ erhält man

$$\overline{\Psi}\,;\,\emptyset\,\vdash \lambda u.(\mu\,(N)\,u)'\,:\,t\approx\mathbf{t}\to t\,\langle\,[\![N]\!],[\![M]\!]\rangle\ \underline{\mathbf{mr}}\,B$$

und

$$\overline{\Psi}\,;\,\emptyset\,\vdash \lambda v.(\mu\,(M)\,v)'\,:\,t\approx\mathbf{f}\to t\,\langle\,[\![N]\!],[\![M]\!]\rangle\ \underline{\mathbf{mr}}\,B$$

so daß (bool $-$ Ind) schließlich das Gewünschte liefert.

$(\mathbf{L}(\tau) - Ind)$ **Fall** $\underline{[\![A]\!] = \star}$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi}\,;\,\emptyset\,\vdash \mu\,(N)\,:\,\forall x^{\mathbf{L}(\tau)}\forall y_1^{\diamond}\forall y_2^{\tau}.\varepsilon\ \underline{\mathbf{mr}}\,A\to\varepsilon\ \underline{\mathbf{mr}}\,A[\mathbf{cons}_\tau\,y_1y_2x/x]$$

$(\mathbf{L}(\tau) - Ind)$ liefert daraus

$$\overline{\Psi},\Gamma\,;\,\emptyset\,\vdash t\,\{\mu\,(N)\}\,:\,\varepsilon\ \underline{\mathbf{mr}}\,A[\mathbf{nil}_\tau/x]\to\varepsilon\ \underline{\mathbf{mr}}\,A[t/x]$$

Fall <u>sonst</u> Nach Induktionsvoraussetzung ist

$$\overline{\Psi}\,;\,\emptyset\,\vdash \mu\,(N)\,:\,\forall x^{\mathbf{L}(\tau)}\forall y_1^{\diamond}\forall y_2^{\tau}\forall z^{[\![A]\!]}.z\ \underline{\mathbf{mr}}\,A\to [\![N]\!]\ y_1y_2z\ \underline{\mathbf{mr}}\,A[\mathbf{cons}_\tau\,y_1y_2x/x]$$

Die zu zeigende Formel $t\,\{\,[\![N]\!]\}\ \underline{\mathbf{mr}}\,(A[\mathbf{nil}_\tau/x]\to A[t/x])$ wird durch $(\mathbf{L}(\tau) - Ind\quad t)$ gezeigt.

Es ist also ein Beweis anzugeben für

$$\begin{array}{rll} & \forall^* x^{\mathbf{L}(\tau)}\forall y_1^{\diamond}\forall y_2^{\tau}. & x\,\{\,[\![N]\!]\}\ \underline{\mathbf{mr}}\,(A[\mathbf{nil}_\tau/x]\to A) \\ & \to & \mathbf{cons}_\tau\,y_1y_2x\,\{\,[\![N]\!]\}\ \underline{\mathbf{mr}}\,(A[\mathbf{nil}_\tau/x]\to \\ & & \qquad A[\mathbf{cons}_\tau\,y_1y_2x/x]) \\ = & \forall^* x^{\mathbf{L}(\tau)}\forall y_1^{\diamond}\forall y_2^{\tau}. & (\forall z^{[\![A]\!]}.z\ \underline{\mathbf{mr}}\ A[\mathbf{nil}_\tau/x]\to x\,\{\,[\![N]\!]\}\,z\ \underline{\mathbf{mr}}\,A) \\ & \to & (\forall z^{[\![A]\!]}.z\ \underline{\mathbf{mr}}\ A[\mathbf{nil}_\tau/x]\to \\ & & \underbrace{\mathbf{cons}_\tau\,y_1y_2x\,\{\,[\![N]\!]\}\,z}_{[\![N]\!]\,y_1y_2(x\{\,[\![N]\!]\,\}z)}\ \underline{\mathbf{mr}}\,A[\mathbf{cons}_\tau\,y_1y_2x/x]) \end{array}$$

Ein möglicher Beweis ist $\lambda^* x^{\mathbf{L}(\tau)}\lambda y_1^{\diamond}\lambda y_2^{\tau}\lambda u\lambda z^{[\![A]\!]}\lambda v^{z\ \underline{\mathbf{mr}}\,A}\ \mu\,(N)\,xy_1y_2(x\,\{\,[\![N]\!]\}\,z)(uvz)$, wobei u eine Annahmevariable vom Typ $\forall z.z\ \underline{\mathbf{mr}}\,A[\mathbf{nil}_\tau/x]\to x\,\{\,[\![N]\!]\}\,z\ \underline{\mathbf{mr}}\,A$.

Außerdem ist offenbar $\mathbf{nil}_\tau\ \{\,[\![A]\!]\}\ \underline{\mathbf{mr}}\,(A[\mathbf{nil}_\tau/x]\to A[\mathbf{nil}_\tau/x])$ beweisbar.

$(\mathbf{N}-Ind)$ **Fall** $[\![A]\!]=\star$ Nach Induktionsvoraussetzung ist

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu\,(N_i)\ :\ \forall x^{\mathbf{N}}\forall y^{\diamond}.\varepsilon\,\underline{\mathbf{mr}}\,A\to\varepsilon\,\underline{\mathbf{mr}}\,A[\mathbf{S}_i yx/x]$$

so daß $(\mathbf{N}-Ind)$ hieraus unmittelbar die Behauptung liefert.

Fall sonst Nach Induktionsvoraussetzung ist

$$\overline{\Psi}\ ;\ \emptyset\ \vdash \mu\,(N_i)\ :\ \forall x^{\mathbf{N}}\forall y^{\diamond}\forall z^{[\![A]\!]}.z\,\underline{\mathbf{mr}}\,A\to[\![N]\!]\,yz\,\underline{\mathbf{mr}}\,A[\mathbf{S}_i yx/x]$$

Die zu beweisende Formel $t\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\ \underline{\mathbf{mr}}\,(A[0/x]\to A[t/x])$ wird durch $(\mathbf{N}-Ind)$ gezeigt. Es sind also Beweise anzugeben für:

$$\begin{array}{rll} & \forall^* x^{\mathbf{N}}\forall y^{\diamond}. & x\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\ \underline{\mathbf{mr}}\,(A[0/x]\to A) \\ & \to & \mathbf{S}_i yx\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\ \underline{\mathbf{mr}}\,(A[0/x]\to A[\mathbf{S}_i yx/x]) \\ = & \forall^* x^{\mathbf{N}}\forall y^{\diamond}. & (\forall z^{[\![A]\!]}.z\,\underline{\mathbf{mr}}\,A[0/x]\to x\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\,z\,\underline{\mathbf{mr}}\,A) \\ & \to & (\forall z^{[\![A]\!]}.z\,\underline{\mathbf{mr}}\,A[0/x]\to \underbrace{\mathbf{S}_i yx\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\,z}_{[\![N_i]\!]\,y(x\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}z)}\ \underline{\mathbf{mr}}\,A[\mathbf{S}_i yx/x]) \end{array}$$

Sei u eine Annahmevariable des Typs $\forall z.z\,\underline{\mathbf{mr}}\,A[0/x]\to x\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\,z\,\underline{\mathbf{mr}}\,A$ und v eine Annahmevariable des Typs $z\,\underline{\mathbf{mr}}\,A[0/x]$. Dann ist uzv ein Beweis von $x\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\,z\,\underline{\mathbf{mr}}\,A$. Also ist $\mu\,(N_i)\,xy(x\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\,z)(uzv)$ ein Beweis von $[\![N_i]\!]\ y(x\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\,z)\,\underline{\mathbf{mr}}\,A[\mathbf{S}_i yx/x]$ aus $\overline{\Psi},x,y,z,u,v$. Daraus folgt mit $(\to^+\ v)$, $(\forall^+ z)$, $(\to^+\ u)$, $(\forall^+ y)$ und $(\forall^{*+} x)$ das Gewünschte.

Außerdem ist offenbar $0\,\{\,[\![N_0]\!]\,,[\![N_1]\!]\,\}\ \underline{\mathbf{mr}}\,(A[0/x]\to A[0/x])$ beweisbar.

□

Korollar 2.6.3 (Auswahl). *Ist* $\forall x^{\tau}\exists^* y^{\rho}A$ *mit* $[\![A]\!]=\star$ *geschlossen beweisbar und ist ferner* $(\varepsilon\,\underline{\mathbf{mr}}\,A)\to A$ *beweisbar (etwa* A *atomar), so ist auch* $\exists^* y^{\tau\to\rho}\forall x^{\tau}A[yx/y]$ *beweisbar.*

Beweis. Sei N ein Beweis von $\forall x^{\tau}\exists^* y^{\rho}A$. Nach Satz 2.6.2 ist $\mu\,(N)$ ein Beweis von

$$\begin{aligned} &[\![N]\!]^{\tau\multimap\rho}\ \underline{\mathbf{mr}}\,\forall x^{\tau}\exists^* y^{\rho}A \\ =&\forall x^{\tau}.\,[\![N]\!]\ x\,\underline{\mathbf{mr}}\,\exists^* y^{\rho}A \\ =&\forall x^{\tau}.\varepsilon\,\underline{\mathbf{mr}}\,A[\,[\![N]\!]\ x/y] \end{aligned}$$

Wegen $(\varepsilon\,\underline{\mathbf{mr}}\,A)\to A$ folgt weiter

$$\forall x^{\tau}.A[\,[\![N]\!]\ x/y]$$

wegen $\mathrm{FV}\,(\,[\![N]\!])=\emptyset$ nach Satz 2.6.1 folgt hieraus mit Existenzeinführung $(\boxtimes_{y^{\tau\multimap\rho},\forall xA[yx/y]}\ [\![N]\!])$

$$\exists^* y^{\tau\multimap\rho}\forall x^{\tau}A[yx/y]$$

□

Korollar 2.6.4 (starke Auswahl). *Ist* $\forall^* x^\tau \exists^* y^\rho A$ *mit* $[\![A]\!] = \star$ *(geschlossen) beweisbar und ist ferner* $(\varepsilon \,\underline{\mathbf{mr}}\, A) \to A$ *beweisbar, so ist auch* $\exists^* y^\rho \forall x^\tau A$ *beweisbar.*

Beweis. Sei N ein Beweis von $\forall^* x^\tau \exists^* y^\rho A$. Nach Satz 2.6.2 ist $\mu(N)$ ein Beweis von

$$\begin{aligned} & [\![N]\!]^\rho \,\underline{\mathbf{mr}}\, \forall^* x \exists^* y A \\ =& \forall x.\, [\![N]\!] \,\underline{\mathbf{mr}}\, \exists^* y A \\ =& \forall x. \varepsilon \,\underline{\mathbf{mr}}\, A[\,[\![N]\!]/y] \end{aligned}$$

Wegen $(\varepsilon \,\underline{\mathbf{mr}}\, A) \to A$ folgt weiter

$$\forall x. A[\,[\![N]\!]/y]$$

wegen FV $([\![N]\!]) = \emptyset$ nach Satz 2.6.1 folgt hieraus mit Existenzeinführung $(\boxtimes_{y^{\tau \multimap \rho}, \forall x A}\, [\![N]\!])$

$$\exists^* y \forall x A$$

□

2.7 Beipiel

Beispiel 2.7.1 (append). Das Prädikat $\mathtt{APPEND}(l, l', l'')$ möge bedeuten, daß l'' aus l und l' durch Verketten entsteht und sei wie üblich definiert. Genauer seien $\blacksquare_{\text{app},1}$ und $\blacksquare_{\text{app},2}$ die Allabschlüsse von

$$\begin{aligned} & \mathtt{APPEND}(\mathbf{nil}_\tau, l^{\mathbf{L}(\tau)}, l^{\mathbf{L}(\tau)}) \\ \mathtt{APPEND}(l^{\mathbf{L}(\tau)}, l'^{\mathbf{L}(\tau)}, l''^{\mathbf{L}(\tau)}) \to & \mathtt{APPEND}(\mathbf{cons}_\tau x^\diamond y^\tau l, l', \mathbf{cons}_\tau x y l'') \end{aligned}$$

Wir zeigen die Terminierung dieses Logikprogramms. Genauer wird ein geschlossener Beweis von $\forall l'^{\mathbf{L}(\tau)}, l^{\mathbf{L}(\tau)} \exists^* l''^{\mathbf{L}(\tau)} \mathtt{APPEND}(l, l', l'')$ angegeben:

> Sei also l' gegeben; zeige die Behauptung durch Induktion nach l. Im Schrittfall schließe mit $\blacksquare_{\text{app},2}$ aus der Induktionsvoraussetzung. Der Anfangsfall folgt aus $\blacksquare_{\text{app},1}$.

Formalisiert lautet der Beweis:

$$\begin{aligned} \lambda l', l \quad l\{ & \lambda^* l \lambda x^\diamond, z^\tau, u^{\exists^* l''\, \mathtt{APPEND}(l,l',l'')}\, u(\lambda l'', v^{\mathtt{APPEND}(l,l',l'')} \\ & \boxtimes_{l'', \mathtt{APPEND}(l,l',l'')}\, (\mathbf{cons}_\tau x z l'')(\blacksquare_{\text{app},2} l l' l'' x z v))\} \\ & (\boxtimes_{l'', \mathtt{APPEND}(\mathbf{nil}_\tau, l', l'')} l' \blacksquare_{\text{app},1}) \end{aligned}$$

Damit ergibt sich als extrahiertes Programm:

$$\begin{aligned} \lambda l', l \quad l\{ & \lambda x^\diamond, z^\tau, x_u^{\mathbf{L}(\tau)} \quad (\lambda l'' \\ & (\lambda y^{\mathbf{L}(\tau)}\, y)(\mathbf{cons}_\tau x z l''))x_u\} \\ & ((\lambda y^{\mathbf{L}(\tau)}\, y) l') \end{aligned}$$

Bemerkung 2.7.2 (Ausblick). Das hier aufgeführte Beispiel läßt sich noch ganz gut "von Hand" bearbeiten. Für größere, "realistischere" Beispiele werden die Beweise jedoch schnell umfangreicher, so daß maschinelle Unterstützung angeraten scheint.

Desweiteren scheint es hilfreich, die im folgenden Kapitel vorgestellten Erweiterungen des zu Grunde liegenden Termsystems in geeigneter Weise zu reflektieren. Man kann etwa hoffen, daß sich nach Einführung von "Divide & Conquer–Induktion" (welche durch Divide & Conquer–Rekursion gemäß Satz 3.2.13 realisiert wird) eine Variante des Warshall–Algorithmuses als extrahiertes Programm ergibt [6]: Statt der üblichen Formulierung "zu zwei Punkten gibt es entweder keine Verbindung oder einen zyklenfreien Weg" könnte etwa eine Aussage wie "zu zwei Punkten und einer Liste von erlaubten Zwischenpunkten sind diese entweder nicht verbunden (und es gibt die unveränderte Liste), oder es gibt eine Verbindung und die Liste der nicht benötigten Zwischenpunkte" bewiesen werden.

3 Erweiterungen von System $\mathcal{C}$

In diesem Kapitel werden Erweiterungen von Hofmanns System $\mathcal{C}$ vorgestellt. Die Beweise, daß auch diese Erweiterungen nicht aus `Ptime` herausführen, verlaufen analog zu denen im ersten Kapitel, was die Flexibilität der gewählten Methode aufzeigt.

Ferner können zu diesen erweiterten Systemen in naheliegender Weise ensprechende Arithmetiksysteme (analog zu dem im zweiten Kapitel vorgestellten) definiert werden. Auf diese Weise stehen die vorgestellten Programmierprinzipien (ganz `Ptime` und "Divide- & Conquerrekursion") auch im Kontext beweisbar korrekter Programme mit beweisbaren Laufzeitschranken zur Verfügung. Dies im Detail auszuführen würde jedoch den Rahmen dieser Diplomarbeit sprengen.

3.1 $\mathcal{C}^{++}$ als Erweiterung auf ganz Ptime, Typ ι

Die Einschränkung auf nicht-wachsende Funktionen mag auf den ersten Blick etwas restriktiv erscheinen. Da die Länge jedoch der Zahl der freien Variablen entspricht, kann man durch geeignete Verwendung des Typs $\diamond$ entscheiden, wie die Länge gezählt werden soll (vgl. dazu Lemma 1.3.3; Seite 13). Ebenso kann durch Nichtverwendung dieses Typs entschieden werden, daß gewisse Längen nicht zu zählen sind. Die prinzipielle Mächtigkeit dieses Ansatzes soll in diesem Abschnitt demonstriert werden: es wird eine Erweiterung $\mathcal{C}^{++}$ von Hofmanns System $\mathcal{C}$ vorgestellt, in dem die definierbaren Funktionen vom Typ $\mathbf{N} \multimap \iota$ gerade die polynomialzeitberechenbaren Funktionen sind. Es bleibt jedoch anzumerken, daß dies mehr eine Demonstration der prinzipiellen Mächtigkeit als eine wirkliche Weiterentwicklung ist, da die ursprüngliche Idee von System $\mathcal{C}$, nämlich Algorithmen auf natürliche Weise definieren zu können, dabei verloren geht. Satz 3.1.12 wird nämlich durch Simulation von Turingmaschinen bewiesen, was für viele Algorithmen *nicht* die natürliche Art der Definition ist.

Die Hauptidee, nämlich Modalitäten in Form von verschiedenen Arten "natürlicher Zahlen" darzustellen, findet sich bereits bei Leivant und Marion [14]. Die Idee, Turingmaschinen zu simulieren, stammt direkt von Hofmann [11].

Zunächst wird das erweiterte Termsystem definiert. Die Hauptidee dabei ist, die "Längenprobleme" (im Sinne Beschränkung auf nicht-wachsende Terme) durch die Hinzunahme eines Typs ι von binären Wörtern "ohne Länge" zu lösen.

Definition 3.1.1 (Termsystem $\mathcal{C}^{++}$). Das Termsystem $\mathcal{C}^{++}$ entsteht aus dem in den Abschnitten 1.1 und 1.2 beschriebenen Termsystem $\mathcal{C}$ durch

- Hinzunahme eines neuen Grundtyps ι

- Hinzunahme folgender Konstanten mit den angegebenen Typen

$$\begin{array}{ll}
\circ & \iota \\
\mathfrak{s}_0 & \iota \multimap \iota \\
\mathfrak{s}_1 & \iota \multimap \iota \\
\mathbf{p} & \iota \multimap \iota \\
\mathbf{iszero} & \iota \multimap (\mathrm{bool} \otimes \iota) \\
\mathbf{head} & \iota \multimap (\mathrm{bool} \otimes \iota)
\end{array}$$

- Erweiterung der Relation $\mapsto$ um folgende Klauseln

$$\begin{array}{l}
\mathbf{p}\circ \mapsto \circ \\
\mathbf{p}(\mathfrak{s}_i t^\iota) \mapsto t^\iota \\
\mathbf{iszero}\circ \mapsto \otimes_{\mathrm{bool},\iota}\mathbf{t}\circ \\
\mathbf{iszero}(\mathfrak{s}_i t^\iota) \mapsto \otimes_{\mathrm{bool},\iota}\mathbf{f}(\mathfrak{s}_i t^\iota) \\
\mathbf{head}\circ \mapsto \otimes_{\mathrm{bool},\iota}\mathbf{f}\circ \\
\mathbf{head}(\mathfrak{s}_0 t^\iota) \mapsto \otimes_{\mathrm{bool},\iota}\mathbf{f}(\mathfrak{s}_0 t^\iota) \\
\mathbf{head}(\mathfrak{s}_1 t^\iota) \mapsto \otimes_{\mathrm{bool},\iota}\mathbf{t}(\mathfrak{s}_1 t^\iota)
\end{array}$$

Die Definitionen für die Relationen $\Gamma \vdash t^\tau$ und $t \to t'$ seien wörtlich übernommen, ebenso die Schreibweise t^τ und der Begriff "fast geschlossen". Dann können mit nahezu wörtlichen Beweisen die grundlegenden Aussagen bewiesen werden:

Proposition 3.1.2. *Proposition 1.1.6, sowie die Lemmata 1.1.7, 1.1.8, 1.2.5 und 1.2.6 gelten auch für $\mathcal{C}^{++}$.*

Um Proposition 1.2.8 entsprechend erweitern zu können, wird zunächst definiert, was die "Werte" vom Typ ι sein sollen: Numerale, ähnlich wie in Definition 1.2.1. Da der Begriff "Numeral" jedoch schon belegt ist, wird der Begriff "kurze Numerale" verwendet, um daran zu erinnern, daß diese Numerale keine "Länge" im Sinne von freien Variablen haben.

Definition 3.1.3 (kurze Numerale). *Kurze Numerale* sind Terme der Form $\mathfrak{s}_{i_1}(\ldots(\mathfrak{s}_{i_n}\circ))$.

Proposition 3.1.4. *Jeder normale, fast geschlossene Term vom Typ ι ist ein kurzes Numeral. Jeder normale, fast geschlossene Term vom Typ* $\mathbf{N}$ *ist ein Numeral. Jeder normale, fast geschlossene Term vom Typ* $\mathbf{L}(\tau)$ *ist eine Liste. Jeder normale, fast geschlossene Term vom Typ* bool *ist* $\mathbf{t}$ *oder* $\mathbf{f}$*. Jeder normale, fast geschlossene Term vom Typ $\diamond$ ist eine Variable.*

Beweis. Im Beweis von Proposition 1.2.8 sind lediglich folgende Fälle zu ergänzen:

$\circ\vec{t}$ Dann ist $\|\vec{t}\| = 0$, da es keine Eliminationsregel für ι gibt.

$\mathfrak{s}_i\vec{t}$ Dann ist $\|\vec{t}\| = 1$ und t_1 ist nach Induktionsvoraussetzung ein kurzes Numeral.

$\mathbf{p}\vec{t}$ Dann ist $\|\vec{t}\| \geq 1$, t_1 nach Induktionsvoraussetzung ein kurzes Numeral und der Term damit also nicht normal.

$\mathbf{iszero}\vec{t}$ ebenso.

$\mathbf{head}\vec{t}$ ebenso.

□

Bei der Definition der Termlänge (im Sinne von "Zahl der Zeichen") kommt wieder die Motivation zum Tragen, daß bei allen "einfachen" Reduktionen, also allen Reduktionen außer der Induktion, der reduzierte Term kürzer sein soll als der ursprüngliche. Bei den Reduktionsregeln für **iszero** und **head** scheinen aber neue Zeichen hinzuzukommen. Dieses Problem kann dadurch gelöst werden, daß man diese Konstanten als besonders groß ansieht.

Definition 3.1.5. Der Begriff der Termlänge für $\mathcal{C}^{++}$ wird (gegenüber Definition 1.3.1, Seite 13) dahingehend verändert, daß

$$|\mathbf{iszero}| = |\mathbf{head}| = 3$$

Proposition 3.1.6. *Auch in $\mathcal{C}^{++}$ gilt: $\Gamma_1 \vdash t^\tau$, $\Gamma_2 \vdash s^\rho$; Γ_1, Γ_2 disjunkt $\Longrightarrow |t[s/x]| \leq |t| + |s|$*

Beweis. wörtlich wie Beweis von Lemma 1.3.2. □

Lemma 3.1.7. *Auch in $\mathcal{C}^{++}$ gilt: $\Gamma \vdash t^\diamond \Longrightarrow \Gamma \neq \emptyset$*

Beweis. Im Beweis von Lemma 1.3.3 sind lediglich folgende Fälle zu ergänzen:

- Fälle $\circ\vec{t}$, $\mathfrak{s}_i\vec{t}$ und $\mathbf{p}\vec{t}$: Terme dieser Bauart haben *nie* den Typ $\diamond$, da es für ι keine Eliminationsregel gibt.
- Fälle $\mathbf{iszero}\vec{t}$ und $\mathbf{head}\vec{t}$: Da t den Typ $\diamond$ hat muß $\|\vec{t}\| \geq 2$ gelten. Dann kann die Induktionsvoraussetzung auf den *kürzeren*(!) Term $\otimes_{\mathsf{bool},\iota}\mathbf{t} \circ t_2 \ldots t_n$ angewandt werden. Da dieser Term nicht mehr freie Variablen entält, folgt die Behauptung.

□

Die polynomiale Länge $\vartheta(t)$ eines Terms t sei wie in Definition 1.3.9 definiert. Wie in Lemma 1.3.10 sieht man dann $\Gamma_1 \vdash t^\tau$,$\Gamma_2 \vdash s^\rho$; Γ_1, Γ_2 disjunkt $\Longrightarrow \vartheta(t[s/x]) \preccurlyeq \vartheta(t) + \vartheta(s)$. Auch der Rest macht keine Probleme:

Satz 3.1.8. *Auch für das Termsystem $\mathcal{C}^{++}$ gilt:*

$\Gamma \vdash t^\tau$; $\natural \in \mathbb{N}$, $\natural \geq |\mathrm{FV}(t)|$. Dann gibt es $t = t_0 \to t_1 \to \ldots \to t_k$ mit t_k normal und $\vartheta(t_k)(\natural) + |t_k| + k \leq \vartheta(t)(\natural) + |t|$

Beweis. Wie der Beweis zu Satz 1.3.11. Die zusätzlichen Redexe können jederzeit reduziert werden, etwa nach jedem Schritt der vorgestellten speziellen Reduktionsstrategie. Da dabei $\vartheta(t)$ nicht wächst und $|t|$ echt abnimmt, funktioniert der Induktionsbeweis jeweils genauso. ∎

Korollar 3.1.9 (Korrektheit von $\mathcal{C}^{++}$). $\emptyset \vdash t^{\vec{\mathbf{N}} \multimap \iota}$. *Dann bezeichnet t eine Funktion, die in polynomial vielen Reduktionsschritten berechnet werden kann.*

Beweis. Seien $\vec{\nu}$ Eingabenumerale. Dann ist $\vartheta(t\vec{\nu}) = \vartheta(t)$, da $\vartheta(\vec{\nu}) = 0$. Eine obere Schranke für die Zahl der Schritte ist also $\vartheta(t)\,(|\mathrm{FV}(\vec{\nu})|) + |t| + |\vec{\nu}|$ und damit polynomial in der Länge der Eingabe $\vec{\nu}$. ∎

Die folgenden Aussagen sind Modifaktionen und einfache Erweiterungen der Aussagen aus [11, 4.3].

Lemma 3.1.10. *Ist $\mathfrak{f}\colon \tau \otimes \mathbf{L}(\rho) \multimap \tau \otimes \mathbf{L}(\rho)$ in $\mathcal{C}$ bzw. in $\mathcal{C}^{++}$ geschlossen definierbar, so auch eine Funktion $\mathfrak{f}^\sharp\colon \tau \otimes \mathbf{L}(\rho) \multimap \tau \otimes \mathbf{L}(\rho)$ mit*

$$\mathfrak{f}^\sharp(a \otimes \ell) = \mathfrak{f}^n(a \otimes \ell)$$

wobei n die Zahl der Elemente von ℓ bezeichnet.

Beweis. Wir argumentieren informell über die definierten Funktionen. Die formale Definition von $\mathfrak{f}^\sharp$ (oder genauer von $t^\sharp$, wenn t ein $\mathfrak{f}$ bezeichnender Term ist) ist dann offensichtlich.

Zunächst kann durch $\mathbf{L}(\rho)^-$ eine Funktion $\mathfrak{g}\colon \mathbf{L}(\rho) \multimap \mathbf{L}(\rho) \multimap \tau \multimap \tau \otimes \mathbf{L}(\rho)$ definiert werden mit

$$\mathfrak{g}(\mathbf{nil}_\rho)(\ell')(b) = b \otimes \ell'$$
$$\mathfrak{g}(\mathbf{cons}_\rho a\ell)\ell' b = \mathfrak{f}(\mathfrak{g}(\ell)(\mathsf{app}(\ell')(\mathbf{cons}_\rho a \mathbf{nil}_\rho)))$$

mit app der Append-Funktion auf Listen. (Diese ist definierbar, siehe etwa Abschnitt 2.7 ab Seite 35.) Durch Induktion über den Aufbau von ℓ (als Liste) sieht man, daß $\mathfrak{g}\ell\ell' b = \mathfrak{f}^n(b \otimes (\mathsf{app}\ell'\ell))$, wobei n die Zahl der Elemente von ℓ ist. Es kann also $\mathfrak{f}^\sharp(a \otimes \ell) = \mathfrak{g}\ell\mathbf{nil}_\rho a$ definiert werden. Alle diese Definitionen sind offenbar linear getypt (auch was die nicht explizit erwähnte $\diamondsuit$'s angeht!). ∎

Proposition 3.1.11. *Ist $\mathfrak{f}\colon \tau \multimap \tau$ geschlossen definierbar, $k \in \mathbb{N}$, so auch $\mathfrak{g}\colon \tau \otimes \mathbf{L}(\rho) \multimap \tau \otimes \mathbf{L}(\rho)$ mit $\mathfrak{g}(a \otimes \ell) = \mathfrak{f}^{n^k}(a) \otimes \ell$, wobei n die Zahl der Elemente von ℓ ist.*

Beweis. (Argumentiere wieder informell.) Definiere zunächst $\tilde{\mathfrak{f}}\colon\ \tau \otimes \mathbf{L}(\rho) \multimap \tau \otimes \mathbf{L}(\rho)$ mit $\tilde{\mathfrak{f}}(a \otimes \ell) = \mathfrak{f}(a) \otimes \ell$. Durch Induktion über k sieht man, daß (mit den Bezeichnungen aus Lemma 3.1.10)

$$\tilde{\mathfrak{f}}^{\overbrace{\sharp\cdots\sharp}^{k}}(a \otimes \ell) = \mathfrak{f}^{n^k}(a) \otimes \ell$$

mit n der Zahl der Elemente von ℓ. ∎

Nach diesen vorbereitenden Überlegungen sind wir jetzt in der Lage, die Vollständigkeit (für `Ptime`) des vorgestellten Systems zu zeigen.

Satz 3.1.12 (Vollständigkeit von $\mathcal{C}^{++}$). *Jede polynomialzeitberechenbare Funktion wird durch einen geschlossenen Term t aus $\mathcal{C}^{++}$ vom Typ $\mathbf{N} \multimap \iota$ denotiert.*

Beweis (Skizze). Ist $\mathfrak{f}$ eine polynomialzeitberechenbare Funktion, so gibt es eine Turingmaschine, die $\mathfrak{f}$ in polynomial vielen Schritten berechnet. Es kann angenommen werden, daß die Turingmaschine über dem Alphabet $\{0, 1\}$ arbeitet. Die Turingmaschine habe $\mathfrak{N}$ Zustände $\{\mathfrak{S}_0, \ldots, \mathfrak{S}_{\mathfrak{N}-1}\}$. Dabei sei $\mathfrak{S}_0$ der Haltezustand und $\mathfrak{S}_1$ der Anfangszustand. Ferner sei n die kleinste natürliche Zahl mit $2^n \geq \mathfrak{N}$.

Eine Konfiguration der Turingmaschine, in der sich bis zum Lesekopf die Zeichen $i_0 \ldots i_k$ (mit Lesekopf auf i_k) und nach dem Lesekopf die Zeichen $j_0 \ldots j_{k'}$ (gefolgt von den noch nicht besuchten Feldern, die mit 0 beschriftet sind) befinden und bei dem die Maschine im Zustand $\mathfrak{S}_m$ ist, wird durch

$$\otimes(\mathfrak{s}_{i_k}(\ldots(\mathfrak{s}_{i_0}\circ)))(\otimes(\mathfrak{s}_{j_0}(\ldots(\mathfrak{s}_{i_{k'}}\circ)))(\otimes t_1(\otimes\ldots(\otimes t_{n-1}t_n))))$$

codiert. Dabei ist jeder der $\vec{t}$ entweder $\mathbf{t}$ oder $\mathbf{f}$ und zwar so, daß dadurch m binär codiert wird. Konfigurationen sind also *geschlossene* Terme vom Typ $\tau = \iota \otimes \iota \otimes \mathrm{bool} \otimes \ldots \otimes \mathrm{bool}$.

Durch $\mathbf{N}$-Elimination mit Ergebnistyp $\iota \otimes \mathbf{N}$ definiert man einen Term, der ein Numeral sowohl in das entsprechende kurze Numeral umwandelt als auch unverändert zurückgibt. Dies ist möglich da die Nachfolger von Numeralen vom Typ ι gebildet werden können, ohne daß dafür freie Variablen benötigt würden. Diesen Term benutzt man um einen Term vom Typ $\mathbf{N} \multimap \mathbf{N} \otimes \tau$ zu definieren, der die Eingabe unverändert zurückliefert und gleichzeitig den Anfangszustand konstruiert.

Die Einschrittübergangsfunktion der Turingmaschine kann ebenfalls durch einen geschlossenen Term definiert werden: Durch entsprechende $\otimes$-Elimination zerlegt man den Zustand in seine Bestandteile, durch Verwendung der *Konstanten* **iszero** und **head** (gefolgt von weiteren $\otimes$-Eliminationen) erhält man zusätzliche Variable vom Typ bool, die angeben, wie das gerade gelesene Zeichen aussieht und ob man sich am Anfang oder "Ende" des Bandes befindet. Durch vollständige Fallunterscheidung (via bool-Elimination) über die Variablen vom Typ bool reduziert sich das Problem darauf, aus dem beiden "Hälften" des Bandes (gegeben durch zwei Variable vom Typ ι, die in *jedem* der Fälle verwendet werden dürfen) den neuen Zustand zu konstruieren. Dies gelingt unter Verwendung der Konstanten $\mathfrak{s}_0$, $\mathfrak{s}_1$ und $\mathbf{p}$, sowie $\otimes$, $\mathbf{t}$ und $\mathbf{f}$.

Geeignetes Iterieren der Einschrittübergangsfunktion mittels Proposition 3.1.11 liefert die Behauptung.

□

3.2 Bäume und "Divide- & Conquer-Rekursion"

In diesem Kapitel wird System $\mathcal{C}$ um den neuen Datentyp "Bäume" erweitert. Dies geschieht in einer Weise, daß Satz 1.3.11 weiterhin gilt. Als Bäume wird hier die allgemeinste Form von Bäumen gewählt: sowohl die Knoten als auch die Blätter tragen Beschriftungen; dabei können diese Beschriftungen von verschiedenem Typ sein.

Neben der ohnehin gegebenen Notwendigkeit, Bäume zuzulassen, damit $\mathcal{C}$ als "vollwertige" "idealisierte Programmiersprache" für nicht-wachsende Ptime Berechnungen betrachtet werden kann, besteht die Hauptmotivation für die Betrachtung von Bäumen darin, daß bei Vorhandensein dieser Bäume "Divide & Conquer"–Rekursion *definiert* werden kann. Mit diesem (in Satz 3.2.13 aufgezeigten) Mechanismus kann also die "natürliche" Definition von durch "Divide & Conquer"–Rekursion definierten Funktionen (wie etwa `quicksort`) in einen etsprechenden Term in (dem erweiterten) System $\mathcal{C}$ übersetzt werden.

Definition 3.2.1 (Erweiterung um Bäume). Das Termsystem der um Bäume erweiterte Fassung von Sytem $\mathcal{C}$ entsteht aus $\mathcal{C}$ durch

- Hinzunahme eines Typs $\mathbf{T}(\tau,\rho)$ von "Bäumen".
- Hinzunahme von Konstanten mit ihren jeweiligen Typen

$$\begin{array}{ll} \mathbf{leaf}_{\tau,\rho} & \rho \multimap \mathbf{T}(\tau,\rho) \\ \mathbf{tree}_{\tau,\rho} & \diamond \multimap \tau \multimap \mathbf{T}(\tau,\rho) \multimap \mathbf{T}(\tau,\rho) \multimap \mathbf{T}(\tau,\rho) \end{array}$$

- Hinzunahme eines neuen Termkonstruktors $\{\cdot,\cdot\}_{\mathrm{T}}$
- Hinzunahme einer neuen Typisierungsregel

$$\frac{\Gamma \vdash t^{\mathbf{T}(\tau,\rho)} \qquad \emptyset \vdash s^{\diamond \multimap \tau \multimap \sigma \multimap \sigma \multimap \sigma} \qquad \emptyset \vdash r^{\rho \multimap \sigma}}{\Gamma \vdash (t\,\{s,r\}_{\mathrm{T}})^{\sigma}} \qquad (\mathbf{T}(\tau,\rho)^{-})$$

Definition 3.2.2 (Bäume). Auf den Termen vom Typ $\mathbf{T}(\tau,\rho)$ wird induktiv (über den Aufbau von Termen) ein Prädikat "Baum" definiert durch

- $\mathbf{leaf}_{\tau,\rho} t^{\rho}$ ist ein Baum
- Sind t_1 und t_2 Bäume, so auch $\mathbf{tree}_{\tau,\rho} d^{\diamond} t^{\tau} t_1 t_2$

Definition 3.2.3 (Reduktionsregeln). Die Reduktionsregeln der um Bäume erweiterten Fassung von Sytem $\mathcal{C}$ sind die von $\mathcal{C}$ sowie die folgenden:

$$\begin{array}{ll} \mathbf{leaf}_{\tau,\rho} t\,\{s,r\}_{\mathrm{T}} \mapsto rt & \\ \mathbf{tree}_{\tau,\rho} dtt_1t_2\,\{s,r\}_{\mathrm{T}} \mapsto sdt(t_1\,\{r,s\}_{\mathrm{T}})(t_2\,\{r,s\}_{\mathrm{T}}) & t_1, t_2 \text{ Bäume} \end{array}$$

Proposition 3.2.4. *Propopsistion 1.1.6, sowie die Lemmata 1.1.7, 1.1.8, 1.2.5 und 1.2.6 gelten auch für die betrachtete Erweiterung.*

Beweis. Wörtlich gleiche Beweise, gegebenenfalls auf triviale Weise um die fehlenden Fälle ergänzt. □

Proposition 3.2.5. *Jeder normale, fast geschlossene Term vom Typ $\mathbf{T}(\tau, \rho)$ ist ein Baum. Jeder normale, fast geschlossene Term vom Typ $\mathbf{N}$ ist ein Numeral. Jeder normale, fast geschlossene Term vom Typ $\mathbf{L}(\tau)$ ist eine Liste. Jeder normale, fast geschlossene Term vom Typ* bool *ist* $\mathbf{t}$ *oder* $\mathbf{f}$. *Jeder normale, fast geschlossene Term vom Typ $\diamond$ ist eine Variable.*

Beweis. Im Beweis zu Proposition 1.2.8 sind lediglich folgende Fälle zu ergänzen:

- Fall $\mathbf{leaf}_{\tau,\rho}\vec{t}$. Damit der Term keinen $\multimap$-Typ hat muß $\|\vec{t}\| \geq 1$ sein. Damit der Term normal ist muß dann aber $\|\vec{t}\| \leq 1$ sein. Also ist t ein Baum.
- Fall $\mathbf{tree}_{\tau,\rho}\vec{t}$. Dann muß $\|\vec{t}\| \geq 4$ sein, damit t keinen $\multimap$-Typ hat. Nach Induktionsvoraussetzung sind dann aber t_3 und t_4 Bäume. Also muß $\|\vec{t}\| \leq 4$ sein, damit t normal ist. Also ist t ein Baum.

□

Definition 3.2.6. Länge eines Terms Definition 1.3.1 wird ergänzt um

$$|\{s, r\}_{\mathrm{T}}| = |r|$$

und Definition 1.3.9 um

$$\vartheta\left(\{s, r\}_{\mathrm{T}}\right) = X \cdot \vartheta\left(s\right) + \left(X + 1\right) \cdot \vartheta\left(r\right) + X \cdot |s| + X \cdot |r|$$

Proposition 3.2.7. *Die Lemmata 1.3.2 und 1.3.3 gelten weiterhin.*

Beweis. Lemma 1.3.2 mit gleichem Beweis. In Lemma 1.3.3 ergänze die Fälle:

- Fall $\mathbf{leaf}_{\tau,\rho}\vec{t}$. Damit t den Typ $\diamond$ hat, muß $\|\vec{t}\| \geq 2$ und t_2 von der Form $t_2 = \{s, r\}_{\mathrm{T}}$ sein. Dann ist $rt_1t_3 \dots t_n$ ein kürzerer(!) Term mit nicht mehr freien Variablen, so daß die Induktionsvoraussetzung die Behauptung liefert.
- Fall $\mathbf{tree}_{\tau,\rho}\vec{t}$. Dann ist $\|\vec{t}\| \geq 5$ und die Induktionsvoraussetzung angewandt auf t_2 liefert die Behauptung.

□

Definition 3.2.8 (Totale Applikation). Sind s und r Terme und ist t ein Baum, so wird durch Induktion über "t Baum" ein Term $@_{s,r}t$ definiert:

$$@_{s,r}(\mathbf{leaf}_{\tau,\rho}t) = rt$$
$$@_{s,r}(\mathbf{tree}_{\tau,\rho}dtt_1t_2) = sd(@_{s,r}t_1)(@_{s,r}t_2)$$

Lemma 3.2.9. *Seien s, r Terme und t ein Baum, so daß $t\,\{s,r\}_{\mathrm{T}}$ getypt im Kontext Γ ist. Dann gilt:*

$$t\,\{s,r\}_{\mathrm{T}} \to^{n+1} @_{s,r}t$$
$$|@_{s,r}t| + n + 1 \leq |t| + n\,|s| + (n+1)\,|r|$$
$$\vartheta\,(@_{s,r}t) \preccurlyeq \vartheta\,(t) + n\vartheta\,(s) + (n+1)\vartheta\,(r)$$

für ein $n \leq |\Gamma|$.

Beweis. Induktion("t Baum")

- Fall $t = \mathbf{leaf}_{\tau,\rho}t'$. Dann $t\,\{s,r\}_{\mathrm{T}} \to rt' = @_{s,r}t$ und $|rt'| + 1 = |r| + |t|$, sowie $\vartheta\,(rt') = \vartheta\,(r) + \vartheta\,(t')$. Also folgt die Behauptung mit $n = 0$.

- Fall $t = \mathbf{tree}_{\tau,\rho}dt't_1t_2$. Nach Induktionsvoraussetzung gibt es n_1 und n_2 mit

$$t_i\,\{s,r\}_{\mathrm{T}} \to^{n_i+1} @_{s,r}t_i$$
$$|@_{s,r}t_i| + n_i + 1 \leq |t_i| + n_i\,|s| + (n_i+1)\,|r|$$
$$\vartheta\,(@_{s,r}t_i) \preccurlyeq \vartheta\,(t_i) + n_i\vartheta\,(s) + (n_i+1)\vartheta\,(r)$$

Dann:

$$\begin{aligned}
\mathbf{tree}_{\tau,\rho}dt't_1t_2\,\{s,r\}_{\mathrm{T}} &\to sdt'(t_1\,\{s,r\}_{\mathrm{T}})(t_2\,\{s,r\}_{\mathrm{T}})\\
&\to^{n_1+1} sdt'(@_{s,r}t_1)(t_2\,\{s,r\}_{\mathrm{T}})\\
&\to^{n_2+1} sdt'(@_{s,r}t_1)(@_{s,r}t_2)
\end{aligned}$$

Ferner:

$$\begin{aligned}
n_1 + n_2 + 3 + |sdt'(@_{s,r}t_1)(@_{s,r}t_2)| = {} & n_1 + n_2 + 3 + |s| + |d| + |t'| + \\
& |(@_{s,r}t_1)| + \\
& |(@_{s,r}t_2)| \\
\leq {} & 1 + |s| + |d| + |t'| + \\
& |t_1| + n_1\,|s| + (n_1+1)\,|r| + \\
& |t_2| + n_2\,|s| + (n_2+1)\,|r| \\
\leq {} & |t| + (n_1+n_2+1)\,|s| + (n_1+n_2+2)\,|r|
\end{aligned}$$

Sowie:

$$\begin{aligned}
\vartheta\,(sdt'(@_{s,r}t_1)(@_{s,r}t_2)) = {} & \vartheta\,(s) + \vartheta\,(d) + \vartheta\,(t') + \\
& \vartheta\,(@_{s,r}t_1) + \vartheta\,(@_{s,r}t_2) \\
\preccurlyeq {} & \vartheta\,(s) + \vartheta\,(d) + \vartheta\,(t') + \\
& \vartheta\,(t_1) + n_1\vartheta\,(s) + (n_1+1)\vartheta\,(r) + \\
& \vartheta\,(t_2) + n_2\vartheta\,(s) + (n_2+1)\vartheta\,(r) \\
\preccurlyeq {} & \vartheta\,(t) + (n_1+n_2+1)\vartheta\,(s) + (n_1+n_2+2)\vartheta\,(r)
\end{aligned}$$

Also folgt die Behauptung mit $n_1 + n_2 + 1$.

□

Korollar 3.2.10. *Satz 1.3.11 gilt auch in dieser Erweiterung.*

Beweis. Für den einzigen noch verbleibenden (nicht-trivialen) Fall $t^{\mathbf{T}(\tau,\rho)}\,\{s,r\}_{\mathrm{T}}\,\vec{t}$ verwende man die spezielle Reduktionsstrategie

$$t\,\{s,r\}_{\mathrm{T}}\,\vec{t} \quad \to^* \quad t'\,\{s,r\}_{\mathrm{T}}\,\vec{t} \quad \to^+ \quad (@_{s,r}t')\vec{t} \quad \to^* \quad \ldots$$

wobei t' eine Normalform von t ist. Lemma 3.2.9 liefert hierfür die richtigen Schranken. $\square$

Es wird nun gezeigt, wie in diesem Termsystem "Divide- & Conquer"–Rekursion definiert werden kann *ohne* neue Typisierungs- oder Reduktionsregeln anzugeben. "Divide- & Conquer"–Rekursion ist also kein eigenes Prinzip, sonder ein aus der natürlichen Eliminationsregel für Bäume abgeleitetes! Der Lesbarkeit halber sind die Beweise (wie auch bei 3.1.10) "informell" angegeben, d.h. es wird von der Semantik der geschlossenen Terme (bezeichnet durch $\mathfrak{f}$ und $\mathfrak{g}$) gesprochen. Trotzdem kann dies als formale Konstruktion von Termen verstanden werden.

Lemma 3.2.11. *Ist* $\mathfrak{f}\colon \tau \multimap \mathbf{L}(\tau) \multimap \tau \otimes \mathbf{L}(\tau) \otimes \mathbf{L}(\tau)$ *geschlossen definierbar, so auch* $\tilde{\mathfrak{f}}\colon \mathbf{L}(\tau) \multimap \mathbf{T}(\tau, \mathbf{L}(\tau))$ *mit*

$$\begin{array}{lcll} \tilde{\mathfrak{f}}(\mathbf{nil}_\tau) & = & \mathbf{leaf}_{\tau,\mathbf{L}(\tau)}\mathbf{nil}_\tau & \\ \tilde{\mathfrak{f}}(\mathbf{cons}_\tau d^\diamond a^\tau \ell) & = & \mathbf{tree}_{\tau,\mathbf{L}(\tau)} d a' \ell_1 \ell_2 & \quad \textit{falls } \mathfrak{f}a\ell = a' \otimes \ell_1 \otimes \ell_2 \end{array}$$

Beweis. Durch $\mathbf{L}(\tau)^-$ definiere man (zunächst) eine Funktion $\mathfrak{g}\colon \mathbf{L}(\tau) \multimap \mathbf{L}(\tau) \times \mathbf{T}(\tau, \mathbf{L}(\tau))$ mit der Semantik $\mathfrak{g}(\ell) = \left\langle \ell, \tilde{\mathfrak{f}}\ell \right\rangle$. $\square$

Lemma 3.2.12. *Ist* $\mathfrak{f}\colon \tau \multimap \mathbf{L}(\tau) \multimap \tau \otimes \mathbf{L}(\tau) \otimes \mathbf{L}(\tau)$ *geschlossen definierbar, so auch eine Funktion* $\bar{\mathfrak{f}}\colon \mathbf{T}(\tau, \mathbf{L}(\tau)) \multimap \mathbf{T}(\tau, \mathbf{L}(\tau))$*, die in einem gegebenen Baum alle Blätter* ℓ *durch* $\tilde{\mathfrak{f}}\ell$ *ersetztt. Dabei ist* $\tilde{\mathfrak{f}}$ *das* $\tilde{\mathfrak{f}}$ *aus Lemma 3.2.11.*

Beweis. $\mathbf{T}(\tau, \mathbf{L}(\tau))^-$ mit Identität für innere Knoten und $\tilde{\mathfrak{f}}$ für Blätter. $\square$

Satz 3.2.13. *Ist* $\mathfrak{f}\colon \tau \multimap \mathbf{L}(\tau) \multimap \tau \otimes \mathbf{L}(\tau) \otimes \mathbf{L}(\tau)$ *geschlossen definierbar, so auch* $\mathfrak{f}^D : \mathbf{L}(\tau) \multimap \mathbf{T}(\tau, \mathbf{L}(\tau))$ *mit*

$$\begin{array}{lcll} \mathfrak{f}^D(\mathbf{nil}_\tau) & = & \mathbf{leaf}_{\tau,\mathbf{L}(\tau)}(\mathbf{nil}_\tau) & \\ \mathfrak{f}^D(\mathbf{cons}_d{}^\diamond a^\tau \ell) & = & \mathbf{tree}_{\tau,\mathbf{L}(\tau)} d a' (\mathfrak{f}^D \ell_1)(\mathfrak{f}^D \ell_2) & \quad \textit{falls } \mathfrak{f}a\ell = a' \otimes \ell_1 \otimes \ell_2 \end{array}$$

Beweis. Sei ℓ das Argument. Dann genügt es, das $\bar{\mathfrak{f}}$ aus Lemma 3.2.12 so oft zu iterieren, wie ℓ Einträge hat; d.h. es genügt ein $\bar{\mathfrak{f}}^\sharp\colon \mathbf{L}(\tau) \multimap \mathbf{T}(\tau, \mathbf{L}(\tau))$ zu definieren mit $\bar{\mathfrak{f}}^\sharp(\ell) = \bar{\mathfrak{f}}^n(\ell)$, wobei n die Zahl der Einträge von ℓ ist. So ein $\bar{\mathfrak{f}}^\sharp$ kann analog zu Lemma 3.1.10 definiert werden. In der Tat ist $\bar{\mathfrak{f}}^\sharp(\ell) = \mathfrak{g}(\ell)(\mathbf{nil}_\tau)$, wobei $\mathfrak{g}\colon \mathbf{L}(\tau) \multimap \mathbf{L}(\tau) \multimap \mathbf{T}(\tau, \mathbf{L}(\tau))$ mit

$$\begin{array}{lcl} \mathfrak{g}(\mathbf{nil}_\tau)(\ell') & = & \bar{\mathfrak{f}}(\mathbf{leaf}_{\tau,\mathbf{L}(\tau)}\ell') \\ \mathfrak{g}(\mathbf{cons}_\tau a\ell)(\ell') & = & \bar{\mathfrak{f}}(\mathfrak{g}(\ell)(\mathsf{app}(\mathbf{cons}_\tau a\mathbf{nil}_\tau)(\ell'))) \end{array}$$

$\square$

"Divide- and Conquer" Rekursion erhält man aus diesem Satz via $\mathbf{T}(\tau, \mathbf{L}(\tau))^-$. Man beachte, daß obige Beweise sämtlich konstruktiv sind. Es wird also auch angegeben, wie aus einer "Dividefunktion" (vom Typ $\tau \multimap \mathbf{L}(\tau) \multimap \tau \otimes \mathbf{L}(\tau) \otimes \mathbf{L}(\tau)$), einer "Conquerfunktion" (vom Typ $\Diamond \multimap \tau \multimap \rho \multimap \rho \multimap \rho$) und einem Anfangsfall (vom Typ ρ, der das Ergebnis für die leere Liste darstellt) eine durch "Divide- & Conquer" Rekursion definierte Funktion in dem betrachteten Termsystem auch tatsächlich definiert werden kann. Es bleibt anzumerken, daß die angegebene Definition nicht optimal ist, was die Komplexität angeht. So kann etwa in Satz 3.2.13 $\mathtt{app}(\mathbf{cons}_\tau a \mathbf{nil}_\tau)(\ell')$ durch $\mathbf{cons}_\tau a\ell'$ ersetzt werden, wenn im *Anfangsfall* eine Funktion `reverse` verwendet wird (die mit linearem Zeitbedarf definiert werden kann!).

Beispiel 3.2.14 (Quicksort). Eine weitere Möglichkeit (als Beispiel 1.4.1), eine Liste zu sortieren kann wie folgt beschrieben werden:

> Die leere Liste ist bereits sortiert. Ansonsten nehme man ein Element der Liste (etwa das erste) und teile den Rest in zwei Teillisten auf, eine mit den kleineren und eine mit den größeren Elementen. Jede dieser Listen sortiere man und anschließend füge man die sortierten Teillisten zusammen.

Hierbei handelt es sich um einen typischen "Divide & Conquer"–Algorithmus: die Liste wird zunächst in *zwei* Bruchstücke zerteilt, die *beide* kleiner sind als die ursprüngliche Liste. Auf *beide* Bruchstücke wird der Algorithmus rekursiv angewandt bevor in einem abschließenden Schritt die beiden Teilergebnisse zu einem Gesamtergebnis zusammengefaßt werden.

Nach obigen Überlegungen genügt es, folgendes anzugeben:

- Eine "Dividefunktion" vom Typ $\tau \multimap \mathbf{L}(\tau) \multimap \tau \otimes \mathbf{L}(\tau) \otimes \mathbf{L}(\tau)$, die eine Liste in zwei Teillisten mit den kleineren und den größeren Elementen zerlegt

$$\begin{array}{l} \lambda x^\tau, l^{\mathbf{L}(\tau)} \quad l\{\lambda y_0^\Diamond, y_1^\tau, p^{\tau\otimes\mathbf{L}(\tau)\otimes\tau} \quad p(\lambda x^\tau, p^{\mathbf{L}(\tau)\otimes\mathbf{L}(\tau)} \quad p(\lambda l', l'' \\ \qquad \lhd_\tau\ xy_1\ \ (\lambda z^{\mathrm{bool}}, p^{\tau\otimes\rho} \qquad p(\lambda x^\tau, y_1^\tau \\ \qquad\qquad z\langle\ \otimes_{\tau,\mathbf{L}(\tau)\otimes\mathbf{L}(\tau)} x(\otimes_{\mathbf{L}(\tau),\mathbf{L}(\tau)}(\mathbf{cons}_\tau y_0 y_1 l')l''), \\ \qquad\qquad\quad \otimes_{\tau,\mathbf{L}(\tau)\otimes\mathbf{L}(\tau)} x(\otimes_{\mathbf{L}(\tau),\mathbf{L}(\tau)} l'(\mathbf{cons}_\tau y_0 y_1 l''))\rangle))))\} \\ \qquad \otimes_{\tau,\mathbf{L}(\tau)\otimes\mathbf{L}(\tau)} x(\otimes_{\mathbf{L}(\tau),\mathbf{L}(\tau)} \mathbf{nil}_\tau \mathbf{nil}_\tau) \end{array}$$

- Eine "Conquerfunktion" vom Typ $\Diamond \multimap \tau \multimap \mathbf{L}(\tau) \multimap \mathbf{L}(\tau) \multimap \mathbf{L}(\tau)$, die zwei sortierte Listen und ein mittleres Element richtig zusammenfügt

$$\lambda x^\Diamond, y^\tau, l'^{\mathbf{L}(\tau)}, l''^{\mathbf{L}(\tau)} \quad \mathtt{app} l'(\mathbf{cons}_\tau x y l'')$$

  Dabei ist `app` die Append-Funktion auf Listen (siehe etwa Beispiel 2.7.1).

- Einem "Anfangsfall" für die leere Liste

$$\mathbf{nil}_\tau$$

A Verwendete Schreibweisen

Grundtypen

Typ	Intendierte Interpretation	Konstruktor-Symbole
$\diamond$	Länge/knappe Resource/leerer Typ/ ...	
$\mathbf{N}$	natürliche Zahlen	0, $\mathbf{S}_0$, $\mathbf{S}_1$
bool	boolesche Werte	$\mathbf{t}$, $\mathbf{f}$
ι	natürliche Zahlen/binäre Worte (keine Längenzählung)	$\circ$, $\mathfrak{s}_0$, $\mathfrak{s}_1$

abgeleitete Typen

Typ	Intendierte Interpretation	Termformer
$\tau \multimap \rho$	Abbildungen	$\lambda x^\tau t^\rho$
$\tau \times \rho$	Paare (im Sinne von `union`)	$\langle \cdot, \cdot \rangle$
$\tau \otimes \rho$	Tensor (Paare/`struct`)	$\otimes_{\tau,\rho}$
$\mathbf{L}(\tau)$	Listen	$\mathbf{nil}_\tau$, $\mathbf{cons}_\tau$
$\mathbf{T}(\tau, \rho)$	Bäume	$\mathbf{leaf}_{\tau,\rho}$, $\mathbf{tree}_{\tau,\rho}$

sonstige Termkonstrukte

$\{\cdot\}$	Termformer für die $\mathbf{L}(\cdot)$-Elimination
$\{\cdot, \cdot\}$	Termformer für die $\mathbf{N}$-Elimination
$\{\cdot, \cdot\}_{\mathrm{T}}$	Termformer für die $\mathbf{T}(\cdot, \cdot)$-Elimination
$\mathbf{p}$	Vorgängerfunktion vom Typ ι
iszero, **head**	Selektionsfunktionen für den Typ ι

Formeln

$\approx$	Gleichheit (Relationssymbol)
$\rightarrow$	wenn ... dann ...
$\wedge$	schwaches und (entspricht Paar)
$\forall$	für alle (Quantor)
$\forall^*$	stark für alle (die quanifizierte Variable wird nicht essentiell benötigt)
$\exists^*$	es gibt (konstruktiver Existenzquantor)
$\vee^*$	konstruktives oder (als Abkürzung verstanden)

Herleitungsterme

$\langle \cdot, \cdot \rangle$	Beweis einer $\wedge$-Formel
$\{\cdot\}$, $\{\cdot, \cdot\}$	Induktion (als Elimination geschrieben)
$\boxtimes_{x,A}$	$\exists^*$-Einführungsaxiom
$\boxminus_{A,x,s,t}$, $\boxminus_t$	Gleichheitsaxiome
$\Psi\ ;\ \Phi \vdash N : A$	N ist Herleitung von A im Kontext Ψ und essentiellen Kontext Φ

Programmextraktion

$t \, \underline{\mathbf{mr}} \, A$	(modifizierte) Realisierbarkeit. $t \, \underline{\mathbf{mr}} \, A$ ist selbst wieder eine Formel
$[\![A]\!]$	extrahierter Typ einer Formel
$\star$	"Untyp". $[\![A]\!] = \star$ heißt, daß A keinen rechnerischen Gehalt hat
$[\![N]\!]$	der extrahierte Term einer Herleitung
ε	der "leere Term" (vom Typ $\star$)
x_u	neue (Objekt)variable zu u
$\widetilde{u}$	Zu u^A gehörige Annahme ($x_u \, \underline{\mathbf{mr}} \, A$)
$\overline{\Psi}$	extrahierter Kontext

Operationen und Relationen auf Termen, Metaterme, ...

$\vec{t}$	endliche Liste von Termen: $t_1, \ldots, t_n$
$\|\cdot\|$	Länge von Termlisten: $\|t_1, \ldots, t_n\| = n$
$t \mapsto t'$	Konversion (äußere Reduktion)
$t \to t'$	Reduktion
$t[s/x]$	Subsitution
$@_{s,r} t$	totale Applikation (bei $\mathbf{T}(\tau, \rho)^-$; Definition 3.2.8)
$\Gamma \vdash t^\tau$	t ist im Kontext Γ getypt vom Typ τ
t^τ	t ist getypter Term vom Typ τ
$\mathrm{FV}\,(t)$	Menge der freien Variablen

Längenmaße auf Termen

$\lvert\cdot\rvert$	Länge im Sinne von Zahl der Zeichen. $\lvert\vec{t}\rvert = \sum \lvert t_i\rvert$
$\vartheta\,(\cdot)$	polynomiale Länge
$f \preccurlyeq g$	natürliche Ordnung auf $\mathbb{N}\,[X]$

Diverses

$\lvert\cdot\rvert$	Mächtigkeit einer Menge. $\lvert\{0,1\}\rvert = 2$
$\mathbb{N}$	Menge der natürlichen Zahlen. $\mathbb{N} = \{0, 1, 2, \ldots\}$
$\mathbb{N}\,[X]$	Polynome über den natürlichen Zahlen in einer Unbestimmten X
`VB!`	Hinweis auf Variablenbedingung (bei arithmetischen Schlußregeln)
π_i	Projektionen. $\pi_i = \lambda x_0, x_1 \, x_i$
$\mathcal{C}$, $\mathcal{C}^{++}$	Termsysteme
ℓHA^ω	Arithmetiksystem

Mitteilungszeichen

Allgemein gilt die Konvention, daß mit $\blacksquare$ jeweils auch $\blacksquare', \blacksquare'', \ldots$ und $\blacksquare_0, \blacksquare_1, \ldots$ Mitteilungszeichen für dieselben Objekte sind.

τ, ρ, σ	Typen
a, t, s, r, h	Terme
d	getypte Terme vom Typ $\diamond$
$\mathfrak{f}, \mathfrak{g}, \mathfrak{h}$	berechenbare Funktionen
x, y, z, l, p, q	(Objekt)variablen
u, v, w	Annahmenvariablen
ν	Numerale
ℓ	Listen
f, g	Polynome (aus $\mathbb{N}[X]$)
$k, n, i, j, m, \mathfrak{N}, \natural$	natürliche Zahlen
Γ	Kontexte (Mengen von Variablen)
A, B, C	Formeln
P, Q	Prädikatsymbole
N, M, L	Herleitungsterme
$\blacksquare$	Axiome (in ℓHA^{ω})
Ψ, Φ	verallgemeinerte Kontexte (Mengen vom Objekt- und Annahmevariablen)

Literatur

[1] Klaus Aehlig and Helmut Schwichtenberg. A syntactical analysis of non-size-increasing polynomial time computation. In *Proceedings of the Fifteenth IEEE Symposium on Logic in Computer Science (LICS '00)*, June 2000. To appear.

[2] Stephen Bellantoni and Stephen Cook. A new recursion-theoretic characterization of the polytime functions. *Computational complexity*, 2:97–110, 1992.

[3] Stephen Bellantoni, Karl-Heinz Niggl, and Helmut Schwichtenberg. Higher type recursion, ramification and polynomial time. To appear: Annals of Pure and Applied Logic.

[4] Ulrich Berger. Program extraction from normalization proofs. In M. Bezem and J.F. Groote, editors, *Typed Lambda Calculi and Applications*, volume 664 of *Lecture Notes in Computer Science*, pages 91–106. Springer Verlag, Berlin, Heidelberg, New York, 1993. `www.mathematik.uni-muenchen.de/~berger/articles/tlca/mr.dvi.Z`.

[5] Ulrich Berger, Martin Hofmann, and Helmut Schwichtenberg. A logic for nonincreasing PTIME computation. Draft, April 1999.

[6] Ulrich Berger, Helmut Schwichtenberg, and Monika Seisenberger. The Warshall Algorithm and Dickson's Lemma: Two Examples of Realistic Program Extraction. Submitted, `www.mathematik.uni-muenchen.de/~schwicht/papers/jar97/jar99.ps.Z`.

[7] Voukko-Helena Caseiro. *Equations for Defining Poly-time Functions*. PhD thesis, University of Oslo, February 1997. `ftp.ifi.uio.no/pub/vuokko/0adm.ps`.

[8] Kurt Gödel. Über eine bisher noch nicht benützte Erweiterung des finiten Standpunkts. *Dialectica*, 12:280–287, 1958.

[9] Martin Hofmann. In-place update with linear types or how to compile functional programms into malloc()-free C. Preprint, `www.dcs.ed.ac.uk/~mxh/malloc.ps.gz`.

[10] Martin Hofmann. Typed lambda calculi for polynomial-time computation. Habilitation thesis, TU Darmstadt, Germany. See `www.dcs.ed.ac.uk/home/mxh/habil.ps.gz`, 1998.

[11] Martin Hofmann. Linear types and non-size-increasing polynomial time computation. In *Proceedings 14'th Symposium on Logic in Computer Science (Lics'99)*, pages 464–473, 1999.

[12] Felix Joachimski and Ralph Matthes. Short Proofs of Normalization for the simply-typed λ-calculus, permutative conversions and Gödel's T. Submitted to the Archive for Mathematical Logic, `www.tcs.informatik.uni-muenchen.de/~matthes/papers/sn.html`, 1998.

[13] Daniel Leivant. Ramified recurrence and computational complexity I: Word recurrence and poly–time. In P. Clote and J. Remmel, editors, *Feasible Mathematics II*, pages 320–343. Birkhäuser, Boston, 1995.

[14] Daniel Leivant and Jean-Yves Marion. Lambda calculus characterization of poly–time. In M. Bezem and J.F. Groote, editors, *Typed Lambda Calculi and Applications*, pages 274–288. Springer Lecture Notes in Computer Science Vol. 664, 1993.

[15] John C. Martin. *Introduction to languages and the theory of computation.* McGraw-Hill, Inc., 1991.

[16] J. van de Pol. *Termination of Higher-order Rewrite Systems.* PhD thesis, Utrecht University, 1996.

[17] Helmut Schwichtenberg. Feasible program from proofs. Submitted, `www.mathematik.uni-muenchen.de/~ schwicht/papers/specker00/fp.ps.Z`, January 2000.

[18] Anne S. Troelstra, editor. *Metamathematical Investigation of Intuitionistic Arithmetic and Analysis*, volume 344 of *Lecture Notes in Mathematics.* Springer Verlag, Berlin, Heidelberg, New York, 1973.