Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Clemens Ballarin

Gleichungslogik und Lambda-Kalkül
Wintersemester 2007/08
Musterlösung zu Übungsblatt 7

Aufgabe 1

Zu zeigen: TRS mit irreduziblen rechten Seiten terminieren.

Fehlanzeige: Man kann nicht einfach argumentieren, daß die Anzahl der Positionen an denen eine Reduktion stattfinden kann in jedem Schritt kleiner wird.

Gegenbeispiel: TRS $f(c) \to d$ und $b \to c$

Betrachte den Startterm $f(b)$. Hier gibt es genau einen Redex, nämlich b. Man erhält: $f(b) \to f(c)$. Jetzt haben wir aber wieder genau einen Redex: $f(c) \to d$

Man sieht an dem Beispiel aber auch, daß der neu entstandene Redex über dem alten liegt. Das führt zu dem nachfolgenden Beweis.

Beweis: Für jeden Term t sind alle in t beginnenden Reduktionssequenzen endlich. Anschaulich gilt dies, da in jedem Term t nur endlich viele Regeln anwendbar sein können. Eine Regelanwendung an Position p in t führt dazu, dass unterhalb von p keine Regeln mehr angewendet werden können (da alle rechten Seiten irreduzibel sind und – da sie keine Variablen enthalten – auch irreduzibel bleiben). Oberhalb von p können sich zwar neue Regelanwendungspunkte ergeben, dies ist aber nur endlich viele Male möglich, bis die Wurzel erreicht ist.

Etwas formaler läßt sich die Multimengenordnung zum Beweis der Termination verwenden:

> Auf der Menge der Positionen in einem Term t besteht die folgende terminierende Ordnung:
>
> $$p_1 < p_2 \;\Leftrightarrow\; p_1 \text{ echt oberhalb von } p_2 \quad (\text{d.h. } \exists x \in \mathbb{N}^+.\ p_1 x = p_2)$$
>
> Sei $P(t)$ die Multimenge der Positionen in t, an denen eine Regel in R anwendbar ist. Eine Position p kommt in $P(t)$ mehrfach vor, wenn an p mehrere Regeln anwendbar sind.
>
> Bei jeder Anwendung einer Regel $l \to r$ an Position p in t_1 mit $t_1 \to t_2$ gilt:
>
> $$P(t_1) >_{\text{mul}} P(t_2), \text{da } P(t_2) = (P(t_1) - X) \cup Y \wedge \forall y \in Y.\ \exists x \in X.\ y < x$$
>
> X ist hierbei die Menge der Positionen, an denen in t_1 Regeln angewendet werden können und in t_2 nicht. X enthält mindestens die Position p.
> Y ist die Menge der Positionen, an denen in t_2 Regeln angewendet werden

können und in t_1 nicht.

Falls Y nicht leer ist, gilt für alle $p' \in Y : p' < p$.

Zu jeder Reduktionsfolge existiert also eine Folge von Mengen, die nach der Multimengenordnung streng monoton abnimmt.
Da diese Multimengenordnung terminiert, müssen auch alle Reduktionsfolgen terminieren.

$\square$

Aufgabe 2

Alle in Σ vorkommenden Funktionen sind unär, und die in $T(\Sigma, V)$ vorkommenden Terme lassen sich mit Wörtern aus Σ^* identifizieren. Dabei wird eine in einem Term vorkommende Variable oder das Konstantensymbol d weggelassen. Dies vereinfacht im Folgenden die Notation.

Sei $v_1 \longrightarrow v_2 \longrightarrow \ldots \longrightarrow v_i \longrightarrow \ldots$ eine Ableitungssequenz. Der Buchstabe a kommt in allen v_i gleich oft vor, und die v_i sind von der Form

$$w_{i0}aw_{i1}a \cdots aw_{i,n-1}aw_{in}, \qquad w_{ij} \in (\Sigma - \{a\}) \tag{*}$$

In einer Ableitung ist die Länge von w_{i0} durch die Länge von w_{00} beschränkt. Damit können die erste und zweite Regel nur endlich oft auf das erste a angewandt werden. Damit ist auch die Länge von w_{i1} beschränkt, und ebenso die von $w_{i2}, \ldots, w_{in}$.

Sei eine Ordnung auf Σ gegeben durch $c > b > a$. Dann ergibt sich die lexikographische Ordnung $>_{\text{lex}}$ auf Σ^*. Diese Ordnung ist nicht notwendig terminierend (z.B. $b >_{\text{lex}} ab >_{\text{lex}} a^2b >_{\text{lex}} \cdots$), sie ist jedoch terminierend bei beschränkter Wortlänge.

Die Ordnung $>_{\text{lex,lex}}$ ist die von $>_{\text{lex}}$ induzierte Ordnung auf n-Tupeln von Worten aus Σ^*. Diese Ordnung können wir auf Worte der Form (*) anwenden, indem wir v_i mit $(w_{i0}, \ldots, w_{in})$ identifizieren. Die Ordnung ist im gegebenen Fall terminierend, da die Längen der w_{ij} beschränkt sind.

Es bleibt zu zeigen, dass die Regeln alle verkleinernd bezüglich $>_{\text{lex,lex}}$ sind. Sei j das Vorkommen von a in (*) oder das Teilwort, auf das die Regel angewandt wird.

$b(a(x)) \longrightarrow a(b^2(c(x)))$: $(\ldots, w_{ij}b, w_{i,j+1}, \ldots) >_{\text{lex,lex}} (\ldots, w_{ij}, b^2cw_{i,j+1}, \ldots)$, da das Wort an Stelle j kürzer und damit kleiner nach $>_{\text{lex}}$ wird.

$c(a(x)) \longrightarrow a(b(c^2(x)))$: $(\ldots, w_{ij}c, w_{i,j+1}, \ldots) >_{\text{lex,lex}} (\ldots, w_{ij}, bc^2w_{i,j+1}, \ldots)$, ebenso.

$c(b(x)) \longrightarrow b(c(x))$: $(\ldots, w_{ij}, \ldots) >_{\text{lex,lex}} (\ldots, w_{i+1,j}, \ldots)$, da im Wort an Stelle j ein Vorkommen von cb duch bc ersetzt wird, und das Wort damit kleiner nach $>_{\text{lex}}$ wird.

Aufgabe 3

$$\begin{aligned} \mathcal{A} :\ & A = \mathbb{N} - \{0, 1\} \\ & x \oplus^{\mathcal{A}} y = 2x + y + 1 \\ & x \odot^{\mathcal{A}} y = x \cdot y \end{aligned}$$

$$\left.\begin{array}{l} \odot, \oplus \text{ sind monoton} \\ \text{A abgeschlossen unter } \oplus, \odot \end{array}\right\} \Rightarrow \mathcal{A} \text{ ist monotone Polynominterpretation}$$

Abgeschlossenheit und Monotonie für $\oplus$ und $\odot$ auf A müssen gezeigt werden (einfach). Das Reduktionssystem R terminiert, falls

$$l_1 >_{\mathcal{A}} r_1,\ l_2 >_{\mathcal{A}} r_2,\ \text{d.h. } P_{l_1} >_A P_{r_1},\ P_{l_2} >_A P_{r_2}$$

$$\begin{array}{lclcll} P_{l_1} & = & 2 \cdot (2x + y + 1) + z + 1 & = & 4x + 2y + z + 3 & >_A \\ P_{r_1} & = & 2x + (2y + z + 1) + 1 & = & 2x + 2y + z + 2 & \end{array}$$

$\forall x, y, z \in A.\ P_{l_1}(x, y, z) > P_{r_1}(x, y, z)$

$$\begin{array}{lclcll} P_{l_2} & = & x \cdot (2y + z + 1) & = & 2xy + xz + x & >_A \\ P_{r_2} & = & 2xy + xz + 1 & & & \end{array}$$

$\forall x, y, z \in A.\ P_{l_2}(x, y, z) > P_{r_2}(x, y, z)$

Aufgabe 4

Gegenbeispiel: $R = \{s \longrightarrow a, a \longrightarrow b, b \longrightarrow f(b)\}$

Algorithmus beginnt mit dem Aufzählen aller Reduktionssequenzen für die rechte Seite der 1. Regel $s \longrightarrow a$:

$$\begin{array}{ccccccc} a & \longrightarrow & b & \longrightarrow & f(b) & \longrightarrow & \ldots \quad \ldots \\ & & \text{enthält a ?} & & \text{enthält a ?} & & \end{array}$$

Problem: Entscheidungsprozedur terminiert für die Reduktionssequenzen von a nicht.

Lösung: Wie in der Vorlesung besprochen bekommt man ein Entscheidungsverfahren, wenn man von allen rechten Seiten gleichzeitig beginnt und dann Breitensuche (statt Tiefensuche) durch diesen Wald macht.