

Lösungsvorschlag *Termersetzungssysteme* – Blatt 11

Aufgabe 1

a) Auch wenn man nur eine Regel hat, erhält man nach Umbenennen der Variablen eine Überlappung:

$$\begin{aligned} f(f(x)) &\rightarrow g(x) \\ f(\underline{f(y)}) &\rightarrow g(y) \end{aligned}$$

Der Unifikator ist $\sigma = \{y/f(x)\}$. Auf die untere Regel angewendet erhält man den Term $f(f(f(x)))$ welcher mit der ersten Regel zu $g(f(x))$ und mit der zweiten zu $f(g(x))$ ausgewertet. Dies ist also unser kritisches Paar, welches offensichtlich *nicht* zusammenführbar ist, da beide Terme in Normalform sind. Daher ist \mathcal{R}_1 *nicht* konfluent.

b) Hier gibt es drei kritische Paare. Zunächst betrachten wir folgende Überlappung:

$$\begin{aligned} f(x, f(y, z)) &\rightarrow f(f(x, y), z) & \sigma = \{y'/x, z'/f(y, z)\} \\ f(x', \underline{f(y', z')}) &\rightarrow f(f(x', y'), z') \end{aligned}$$

Das führt zu dem Term $f(x', f(x, f(y, z)))$, welcher mit erster und zweiter Regel zum kritischen Paar $\langle f(x', f(f(x, y), z)), f(f(x', x), f(y, z)) \rangle$ führt. Dieses Paar lässt sich aber wieder zusammenführen:

$$\begin{aligned} f(x', f(f(x, y), z)) &\rightarrow f(f(x', f(x, y)), z) \rightarrow f(f(f(x', x), y), z) \\ f(f(x', x), f(y, z)) &\rightarrow f(f(f(x', x), y), z) \end{aligned}$$

Als nächstes betrachten wir die Überlappung zwischen erster und zweiter Regel:

$$\begin{aligned} f(x, \underline{f(y, z)}) &\rightarrow f(f(x, y), z) & \sigma = \{y/e, z/x'\} \\ f(e, x') &\rightarrow x' \end{aligned}$$

Das führt zum Term $f(x, f(e, x'))$, der zu dem kritischen Paar $\langle f(f(x, e), x'), f(x, x') \rangle$ führt. Beide Terme sind in Normalform bzgl. \rightarrow , so dass das Paar nicht zusammenführbar ist. Damit ist \mathcal{R}_2 nicht lokal konfluent und damit auch nicht konfluent. Wir haben allerdings noch ein drittes kritisches Paar, welches aus der Unifikation der ersten und zweiten Regel an der Root-Position entsteht mit $\sigma = \{x/e, x'/f(y, z)\}$. Dies führt zum kritischen Paar $\langle f(y, z), f(f(e, y), z) \rangle$. Dieses lässt sich aber einfach durch Anwenden der zweiten Regel zusammenführen.

c) Das erste kritische Paar bildet sich analog zu b) und lässt sich auch ebenso zusammenführen. Zusätzlich haben wir eine ähnliche Überlappung wie bei b) zwischen erster und zweiter Regel, die zu dem kritischen Paar $\langle f(f(x, x'), e), f(x, x') \rangle$ führt, welches in diesem Fall zusammenführbar ist mit $f(f(x, x'), e) \rightarrow f(x, x')$. Es liegt also ein lokal konfluentes TES vor mit dem Kritische-Paare-Lemma. Mit einer LPOS mit Status $\pi_f = \langle 2, 1 \rangle$ lässt sich die Terminierung von \mathcal{R}_3 beweisen, also liegt mit dem Diamond-Lemma auch Konfluenz vor.

d) Hier haben wir kein kritisches Paar: Wegen der Funktionssymbole könnte es allerhöchstens eine Überlappung auf Root-Ebene geben. Dort unifizieren die Regeln aber nicht, denn

$$\{x =? s^i(x'), s^j(x) =? x'\} \implies \{x =? s^i(x'), s^{i+j}(x') =? x'\}$$

Dies führt zu einem Abbruch mittels Occur-Failure, da $i + j = 0$ wegen $i > 0$ nicht eintreten kann. Außerdem gibt es den Fall, in dem $\text{gt}(x, s^i(x))$ und $\text{gt}(x', s^j(x'))$ bzw. $\text{gt}(s^i(x), x)$ und $\text{gt}(s^j(x'), x')$ für $i \neq j$ unifiziert werden könnten. Mit den Unifikationsregeln ergeben in beiden Fällen das Unifikationsprobleme

$$\{x =? x', s^i(x') =? s^j(x')\}$$

Nach $\min(i, j)$ -maliger Anwendung der Termreduktion erhält man einen Occur-Failure. Damit ist \mathcal{R}_4 lokal konfluent nach dem Kritische-Paare-Lemma. Da es offensichtlich auch fundiert ist, haben wir mit dem Diamond-Lemma auch die Konfluenz gezeigt.

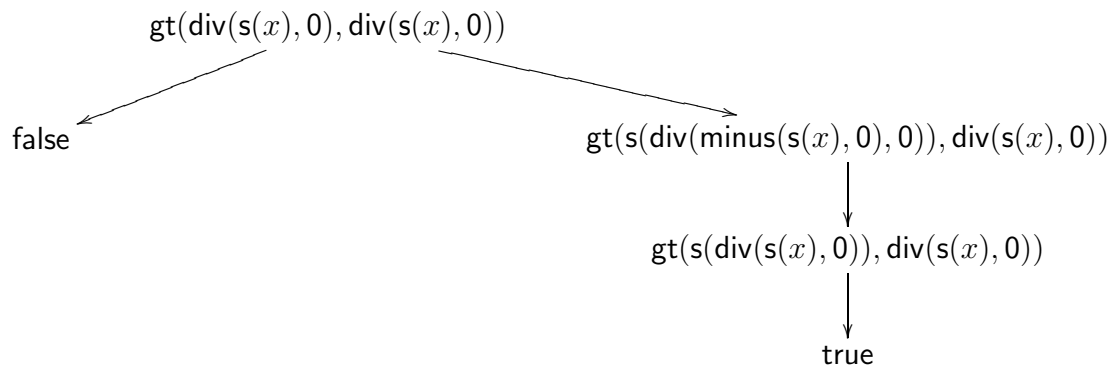
e) Keine linke Seite und auch kein Teilterm einer linken Seite unifiziert mit einer anderen linken Seite. Wir haben also keine kritischen Paare und \mathcal{R}_5 ist mit dem Kritische-Paare-Lemma lokal konfluent. Das TES terminiert allerdings nicht wegen

$$\text{div}(s(x), 0) \rightarrow s(\text{div}(\text{minus}(s(x), 0), 0)) \rightarrow s(\text{div}(s(x), 0))$$

also ist das Diamond-Lemma nicht anwendbar. Allerdings ist das TES *orthogonal*, da wir keine kritischen Paare haben und die Gleichungen auch *linear* sind. Laut Vorlesung folgt aus der Orthogonalität aber die Konfluenz, also ist \mathcal{R}_5 konfluent.

f) Dieses TES hat ebenfalls keine kritischen Paare, da \mathcal{R}_4 und \mathcal{R}_5 jeweils keine kritischen Paare hatten und sich die Regeln der beiden TESe nicht überlappen. Damit ist \mathcal{R}_6 nach dem Kritische-Paare-Lemma lokal konfluent. Das TES ist aber

nicht konfluent, wie folgendes Beispiel zeigt:

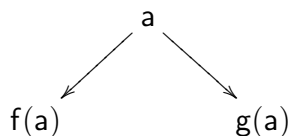


Aufgabe 2

a) Wir betrachten das folgende TES:

$$\mathcal{R} = \{a \rightarrow b, a \rightarrow f(a), a \rightarrow g(a), f(x) \rightarrow g(x), b \rightarrow a\}$$

Dieses TES ist *konfluent*, da man zwei beliebige Terme, die aus diesen Funktionssymbolen bestehen, zunächst stets auf die Form $g^i(a)$ bringen kann und dann mit der Regel $a \rightarrow g(a)$ das i beider Terme gleich gemacht werden kann. Aber das TES ist nicht konfluent bezüglich \xrightarrow{i} , denn wir haben:



Um die Terme nun mit \xrightarrow{i} zusammen zu führen müsste man irgendwann die Regel $f(x) \rightarrow g(x)$ anwenden. Da aber weder a noch irgend ein Term, den man aus a ableiten kann, in Normalform bezüglich \mathcal{R} ist, kann man diese Regel für die \xrightarrow{i} -Auswertung niemals benutzen. Das TES ist also nicht konfluent bezüglich \xrightarrow{i} , obwohl es Konfluent bezüglich \rightarrow ist.

b) Sei $\rightarrow_{\mathcal{R}}$ konvergent, also fundiert und konfluent. Zu zeigen: Auch \xrightarrow{i} ist fundiert und konfluent. Die Fundiertheit von \xrightarrow{i} folgt sofort aus $\xrightarrow{i} \subseteq \rightarrow_{\mathcal{R}}$. Nun zur Konfluenz. Es sei $p \xrightarrow{i^*} s, p \xrightarrow{i^*} t$. Da \xrightarrow{i} terminiert, gibt es Normalformen q_s und q_t mit $s \xrightarrow{i^*} q_s$ und $t \xrightarrow{i^*} q_t$. Aus dem Hilfssatz folgt, dass q_s und q_t auch Normalformen bzgl. $\rightarrow_{\mathcal{R}}$ sind. Da $\rightarrow_{\mathcal{R}}$ konfluent ist und auch $p \xrightarrow{*} q_s, p \xrightarrow{*} q_t$ gilt, müssen die beiden Normalformen q_s und q_t gleich sein. Also lassen sich s und t auch mittels \xrightarrow{i} zusammenführen und die Konvergenz von \xrightarrow{i} ist bewiesen.

Aufgabe 3

a) Wir widmen uns zunächst der Frage, wie genau das Band kodiert wird. Dazu verwenden wir den Konkatenationsoperator $:$ und interpretieren $\mathbf{a} : \mathbf{b} : \mathbf{nil}$ als Zeichenkette \mathbf{ab} . Nun fügen wir Regeln für die Zustandsübergänge ein. Dazu benötigen wir zweistellige Funktionssymbole f_q für jeden Zustand. Das erste Argument codiert den Bandinhalt links und beim Kopf, das zweite Argument codiert den Bandinhalt rechts vom Kopf. Da das Band zwar unendlich ist, aber nur auf endlich vielen Stellen von ε verschiedene Einträge stehen, kommen wir mit endlichen Termen dafür aus: Wir brechen dort ab, wo *nur noch* ε -Einträge kommen. Das endliche Bandalphabet Σ wird durch eine endliche Menge von Konstanten dargestellt, wir wählen einfach $\Sigma^0 = \Sigma$.

Je nach Richtung, in die sich der Kopf bewegt, sehen die Regel leicht verändert aus. Zunächst müssten wir die Möglichkeit haben, einer leeren Liste für die Bandbeschreibung ein ε hinzuzufügen zu können:

$$f_q(xs, \mathbf{nil}) \rightarrow f_q(xs, \varepsilon : \mathbf{nil}) \quad f_q(\mathbf{nil}, ys) \rightarrow f_q(\varepsilon : \mathbf{nil}, ys)$$

Offensichtlich haben wir hier keine Probleme mit der lokalen Konfluenz, da es nur kritische Paare $\langle f_q(\mathbf{nil}, \varepsilon : \mathbf{nil}), f_q(\varepsilon : \mathbf{nil}, \mathbf{nil}) \rangle$ gibt, die offensichtlich zusammengeführt werden können.

Nach links: Sei $\delta(q, \mathbf{a}) = (q', \text{left}, \mathbf{b})$. Dann füge folgende Regel in das TES ein;

$$f_q(\mathbf{a} : l, x : r) \rightarrow f_{q'}(l, \mathbf{b} : x : r)$$

Nach rechts: Sei $\delta(q, \mathbf{a}) = (q', \text{right}, \mathbf{b})$. Dann füge folgende Regel in das TES ein:

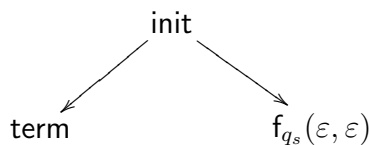
$$f_q(\mathbf{a} : l, x : r) \rightarrow f_{q'}(x : \mathbf{b} : l, r)$$

Um die Terminierung der Turingmaschine bequem anzuzeigen, fügen wir noch die Regel $f_{q_e}(x, y) \rightarrow \mathbf{term}$ hinzu, außerdem noch eine Anfangsregel $\mathbf{init} \rightarrow f_{q_s}(\mathbf{nil}, \mathbf{nil})$. Die Regel $f_{q_e}(x, y) \rightarrow \mathbf{term}$ bildet zwar kritische Paare mit den ersten Regeln für das leere Band, aber diese sind offensichtlich zu dem Term \mathbf{term} zusammenführbar.

Unser TES ist *lokal konfluent*, da es keine weiteren kritische Paare außer den erwähnten gibt: \mathbf{init} überlappt mit nichts. Für eine Überlappung mit einer Regel der Form $f_q(l, r)$ kommt höchstens eine andere Regel mit demselben äußeren Funktionssymbol f_q in Frage. Regeln, die für dasselbe $\mathbf{a} \in \Sigma$ aufgestellt wurden, überlappen nicht, wie weiter oben schon beobachtet wurde. Anderenfalls haben wir

es aber mit Regeln für unterschiedliches a zu tun, also kommen in beiden Regeln unterschiedliche Konstanten an derselben Stelle vor, so dass die Regeln ebenfalls nicht überlappen können. Aus diesen Überlegungen folgt die *lokale Konfluenz*.

b) Klar: Die Turingmaschine terminiert genau dann, wenn wir **init** in endlich vielen Schritten zu **term** reduzieren können. Dies müssen wir mit der lokalen Konfluenz verknüpfen. Dazu fügen wir dem TES einfach die Regel $\text{init} \rightarrow \text{term}$ hinzu. Es gibt dann den Indeterminismus



Wäre die lokale Konfluenz eines TES entscheidbar, könnte man automatisch feststellen, ob es eine Auswertung von $\mathbf{f}_{q_s}(\varepsilon, \varepsilon)$ nach **term** gibt. Dann hätte man aber automatisch entschieden, ob die Turingmaschine terminiert. Dies steht im Widerspruch zur Unentscheidbarkeit des (speziellen) Halteproblems. Also ist die lokale Konfluenz *unentscheidbar*.

c) Zunächst stellen wir fest, dass der bisherige Ansatz nicht *so* verkehrt ist: *Wenn* unser TES terminiert, dann terminiert insbesondere jede von **init** ausgehende Berechnung, also terminiert die Turingmaschine. Wie in der Aufgabenstellung angemerkt, folgt aus der Nichtterminierung des TES aber noch nicht die Nichtterminierung der Turingmaschine. Das Problem sind die nicht-erreichbaren Konfigurationen, die unter Umständen nicht terminieren, obwohl jede erreichbare Konfiguration terminieren würde.

Eine mögliche Lösung dieses Problems ist es, die Simulation der Turingmaschine wie folgt durchzuführen: Beginnend vom Startzustand werten wir nicht beliebig lange aus, sondern maximal i Schritte. Terminiert das TES, terminiert also auch die Turingmaschine. Terminiert es nicht nach i Schritten, beginnen wir erneut beim Startzustand und erlauben diesmal $i + 1$ Schritte. Sollte man nun als Startterm einen unerreichbaren Zustand wählen und eine beliebige Zahl i für die maximale Schrittzahl, so beginnt man mit der Berechnung nach spätestens i Schritten wieder beim Startzustand und berücksichtigt den unerreichbaren Zustand nicht weiter. Terminiert das TES also nicht, so terminiert auch die Turingmaschine nicht.

Formal sieht das TES so aus: Für jede Regel $l \rightarrow r$ aus dem TES von Teil a), füge die Regel

$$\text{trans}(l, y, \mathbf{s}(x)) \rightarrow \text{trans}(r, y, x)$$

ein. x ist also unser Zähler. y brauchen wir, um den maximalen Wert des Zählers zu speichern. Außerdem benötigen wir die Regel

$$\text{trans}(x, y, 0) \rightarrow \text{trans}(f_{q_s}(\text{nil}, \text{nil}), s(y), s(y))$$