Prof. Dr. Jürgen Giesl Florian Frohn

Notes:

- Please solve these exercises in **groups of two**!
- The solutions must be handed in **directly before (very latest: at the beginning of)** the exercise course on Friday, November 20th, 2015, in lecture hall **AH 1**. Alternatively you can drop your solutions into a box which is located right next to Prof. Giesl's office (until the exercise course starts).
- Please write the **names** and **immatriculation numbers** of all (two) students on your solution. Please staple the individual sheets!

Exercise 1 (Examples for Well-foundedness and Confluence): (1 + 1 + 2 = 4 points)

a) Let $\sim \subseteq \mathbb{N}^2$ be the relation such that $x \sim y$ iff $\exists z \in \mathbb{N} \setminus \{1\} : x = y \cdot z$.

Prove or disprove *well-foundedness* and *confluence* of $\sim$.

b) Consider the relation $\supsetneq_A \subseteq \mathcal{P}(A)^2$ for an arbitrary finite set A. Here $\mathcal{P}(A)$ denotes the *power set* of A, i.e., the set of all sets with elements from A. Then $M \supsetneq_A N$ holds iff $M \neq N$ and for all $a \in N$ also $a \in M$ holds. Prove or disprove *well-foundedness* and *confluence* of $\supsetneq_A$.

c) A term s is called *linear* iff for each $x \in \mathcal{V}(s)$ there is one and only one position π such that $s|_\pi = x$. Let $\mathcal{T}_\ell(\Sigma, \mathcal{V})$ be the set of all linear terms over Σ and $\mathcal{V}$ and let $\sqsupseteq \subseteq \mathcal{T}_\ell(\Sigma, \mathcal{V})^2$ be the inverse matching relation, i.e., we have $s \sqsupseteq t$ iff there is a substitution σ such that $s = t\sigma$. We define $s \sqsupset t$ iff $s \sqsupseteq t \wedge t \not\sqsupseteq s$. Prove or disprove *well-foundedness* and *confluence* of $\sqsupset$.

Exercise 2 (Well-foundedness and Confluence): (3 + 2 = 5 points)

Consider an arbitrary binary relation $\to \subseteq A \times A$. Now consider the following three relations:

1. $\to$
2. $\to^+$
3. $\to^{*,\neq}$, where $s \to^{*,\neq} t$ iff $s \to^* t$ and $s \neq t$.

a) Investigate for each pair of these relations if well-foundedness of the first relation implies well-foundedness of the second relation (6 cases). For this, either give a proof or a counterexample.

b) Investigate for each pair of these relations if confluence of the first relation implies confluence of the second relation (6 cases). For this, either give a proof or a counterexample.

Exercise 3 (Programming in Term Rewriting): (2 + 3 = 5 points)

a) Define a term rewrite system $\mathcal{R}$ for reverse, which can be used to reverse a list $[x_1, x_2, \ldots, x_n]$ resulting in the list $[x_n, x_{n-1}, \ldots, x_1]$.

To represent lists, use the two symbols $\mathsf{Nil} \in \Sigma_0$ and $\mathsf{Cons} \in \Sigma_2$. Nil represents the empty list and $\mathsf{Cons}(x, xs)$ represents a list where x is the first list element and xs is the representation of the remaining list elements. Thus, the term $\mathsf{Cons}(x, \mathsf{Cons}(y, \mathsf{Nil}))$ corresponds to the list $[x, y]$. For example, your TRS should allow the reduction $\mathsf{reverse}(\mathsf{Cons}(\mathcal{O}, \mathsf{Cons}(\mathsf{s}(\mathcal{O}), \mathsf{Nil}))) \to_{\mathcal{R}}^* \mathsf{Cons}(\mathsf{s}(\mathcal{O}), \mathsf{Cons}(\mathcal{O}, \mathsf{Nil}))$.

b) Integers can be represented by terms using the function symbols $\mathcal{O} \in \Sigma_0$ and $\mathsf{s}, \mathsf{neg} \in \Sigma_1$, where neg denotes negative numbers. Hence, integers are mapped to terms by the following function $term : \mathbb{Z} \to \mathcal{T}(\{\mathcal{O}, \mathsf{s}, \mathsf{neg}\}, \emptyset)$:

$$term(x) = \begin{cases} \mathsf{s}^{|x|}(\mathcal{O}) & \text{If } x \geq 0 \\ \mathsf{neg}(\mathsf{s}^{|x|}(\mathcal{O})) & \text{Otherwise} \end{cases}$$

Define a term rewrite system $\mathcal{R}$ for plus, which can be used to add integers. For example, your TRS should allow the reduction $\mathsf{plus}(\mathsf{neg}(\mathsf{s}(\mathcal{O})), \mathsf{s}(\mathsf{s}(\mathcal{O}))) \to_{\mathcal{R}}^* \mathsf{s}(\mathcal{O})$. You can assume that the arguments of plus are of the form $\mathsf{s}^x(\mathcal{O})$ or $\mathsf{neg}(\mathsf{s}^x(\mathcal{O}))$, $x \in \mathbb{N}$.

Exercise 4 (Term Rewriting and WHILE Programs): (5 points)

WHILE programs are known to be Turing complete. The goal of this exercise is to prove Turing completeness of term rewriting by simulating WHILE programs with TRSs.

Given a set of variables $\mathcal{V} = \{x_1, \ldots, x_n\}$, the set $\mathcal{W}$ of all WHILE programs operating on $\mathcal{V}$ is the smallest set such that:

- "$x := y + c$"$\in \mathcal{W}$ for all $x, y \in \mathcal{V}, c \in \mathbb{Z}$
- if $w, u \in \mathcal{W}$, then "$w; u$"$\in \mathcal{W}$
- if $w \in \mathcal{W}$, then "WHILE $x \neq 0$ DO w END" $\in \mathcal{W}$ for all $x \in \mathcal{V}$

So we have, e.g., add ="WHILE $y \neq 0$ DO $x = x + 1; y = y + (-1)$ END" $\in \mathcal{W}$.

Given an initial valuation $v : \mathcal{V} \to \mathbb{Z}$, a WHILE program w computes the result $eval(v, w)$ where

$$eval : (\mathcal{V} \to \mathbb{Z}) \times \mathcal{W} \to (\mathcal{V} \to \mathbb{Z})$$

is a partial function which is defined as follows:

$$\begin{aligned} eval(v, x := y + c) &= v' \text{ where } v'(z) = \begin{cases} v(y) + c & \text{if } z = x \\ v(z) & \text{otherwise} \end{cases} \\ eval(v, w; u) &= eval(eval(v, w), u) \\ eval(v, \text{WHILE } x \neq 0 \text{ DO } w \text{ END}) &= \begin{cases} v & \text{if } v(x) = 0 \\ eval(eval(v, w), \text{WHILE } x \neq 0 \text{ DO } w \text{ END}) & \text{otherwise} \end{cases} \end{aligned}$$

So for our example program add, we have $eval(v, add) = v(x) + v(y)$ for all valuations v such that $v(y) \geq 0$.

The program w terminates on the valuation v iff $eval(v, w)$ is defined. So our example program add terminates on the valuation v iff $v(y) \geq 0$.

With $term^{-1}$, we refer to the partial function such that $term^{-1}(term(x)) = x$ (where $term$ is defined as in the previous exercise, i.e., $term^{-1}$ can be used to map terms of the form $\mathsf{s}^x(\mathcal{O})$ or $\mathsf{neg}(\mathsf{s}^x(\mathcal{O}))$ to integers in the expected way). For a given WHILE program w, construct a corresponding term rewrite system $\mathcal{R}$, such that the following holds for all valuations v:

- If w terminates on v (i.e., $eval(v, w)$ is defined), then $\mathsf{start}(term(v(x_1)), \ldots, term(v(x_n)))$ has one and only one $\to_{\mathcal{R}}$-normal form $\mathsf{end}(t_1, \ldots, t_n)$ where $term^{-1}(t_i) = (eval(v, w))(x_i)$ for all $i \in \{1, \ldots, n\}$.
- Otherwise, $\mathsf{start}(term(v(x_1)), \ldots, term(v(x_n)))$ does not have a $\to_{\mathcal{R}}$-normal form.

You do not have to provide a formal proof for your construction, but you should explain it.