Argument Filterings and Usable Rules for Simply Typed Dependency Pairs*
(extended abstract)

Takahito Aoto† Toshiyuki Yamada‡

1 Introduction

Simply typed term rewriting [Yam01] is a framework of higher-order term rewriting without bound variables. The authors extended the first-order dependency pair approach [AG00] to the case of simply typed term rewriting [AY05]. They gave a characterization of minimal non-terminating simply typed terms and incorporated the notions of dependency pairs, dependency graphs, and estimated dependency graphs into the simply typed framework. They extended the subterm criterion [HM04] of first-order dependency pairs and introduced the head instantiation technique to make the simply typed dependency pair method effectively applicable even in the presence of function variables.

In this paper, we incorporate termination criteria using reduction pairs and related refinements into the simply typed dependency pair framework. In particular, we extend the notions of argument filterings [AG00] and usable rules [HM04, TGSK04] of first-order dependency pairs to the case of simply typed term rewriting.

Refinements of dependency pair technique for higher-order systems with bound variables are studied in [Bla06, SK05], and an approach to deal within the framework of first-order dependency pairs is studied in [GTSK05]. In our framework the presence of simple types and higher-order variables/rules are reflected in more specific way comparing with [GTSK05]. On the other hand, since bound variables are not included in our framework, our dependency pair framework is simpler and thus easy to automate compared to the methods in [Bla06, SK05].

2 Preliminaries

A *simple type* is either the *base type* o or a *function type* $\tau_1 \times \cdots \times \tau_n \to \tau_0$. The set of simple types is denoted by ST. The sets of *constants*, *variables*, and *simply typed terms* are denoted by Σ, V, and $\mathrm{T}(\Sigma, V)$, respectively. The *head symbol* of a simply typed term is defined as follows: $\mathrm{head}(a) = a$ for $a \in \Sigma \cup V$;

*The authors thank Jeroen Ketema and the referees for their comments.

†RIEC, Tohoku University, Japan. aoto@nue.riec.tohoku.ac.jp

‡Graduate School of Engineering, Mie University, Japan. toshi@cs.info.mie-u.ac.jp

$\mathrm{head}((t_0\ t_1 \cdots t_n)) = \mathrm{head}(t_0)$. The set $\mathrm{PV}(t)$ of *primary variables* in a term t is defined as follows: $\mathrm{PV}(t) = \emptyset$ if $t \in \Sigma \cup V$; $\mathrm{PV}(t) = \{t_0\} \cup \bigcup_{i>0} \mathrm{PV}(t_i)$ if $t = (t_0\ t_1 \cdots\ t_n)$ and $t_0 \in V$; $\mathrm{PV}(t) = \bigcup_{i \geq 0} \mathrm{PV}(t_i)$ if $t = (t_0\ t_1 \cdots\ t_n)$ and $t_0 \notin V$. Let $\mathcal{R} = \langle \Sigma, R \rangle$ be a *simply typed term rewriting system* (STTRS, for short). The set Σ_{d} of *defined symbols* of $\mathcal{R}$ is defined by $\Sigma_{\mathrm{d}} = \{\mathrm{head}(l) \mid l \to r \in R\}$.

Example 1 (simply typed term rewriting) Let $\mathcal{R} = \langle \Sigma, R \rangle$ be an STTRS where $\Sigma = \{\ \mathsf{0}^{\mathrm{o}},\ \mathsf{s}^{\mathrm{o}\to\mathrm{o}},\ [\,]^{\mathrm{o}},\ :^{\mathrm{o}\times\mathrm{o}\to\mathrm{o}},\ \mathsf{map}^{(\mathrm{o}\to\mathrm{o})\times\mathrm{o}\to\mathrm{o}},\ \circ^{(\mathrm{o}\to\mathrm{o})\times(\mathrm{o}\to\mathrm{o})\to\mathrm{o}\to\mathrm{o}},\ \mathsf{twice}^{(\mathrm{o}\to\mathrm{o})\to\mathrm{o}\to\mathrm{o}}\ \}$, and

$$R = \left\{ \begin{array}{lllll} (1) & \mathsf{map}\ F\ [\,] & \to & [\,] \\ (2) & \mathsf{map}\ F\ (:\ x\ xs) & \to & :\ (F\ x)\ (\mathsf{map}\ F\ xs) \\ (3) & (\circ\ F\ G)\ x & \to & F\ (G\ x) \\ (4) & \mathsf{twice}\ F & \to & \circ\ F\ F \end{array} \right\}.$$

Here is a rewrite sequence of $\mathcal{R}$:

$$\begin{array}{lll} \mathsf{map}\ (\mathsf{twice}\ \mathsf{s})\ (:\ \mathsf{0}\ [\,]) & \to_{\mathcal{R}} & \mathsf{map}\ (\circ\ \mathsf{s}\ \mathsf{s})\ (:\ \mathsf{0}\ [\,]) \\ & \to_{\mathcal{R}} & :\ ((\circ\ \mathsf{s}\ \mathsf{s})\ \mathsf{0})\ (\mathsf{map}\ (\circ\ \mathsf{s}\ \mathsf{s})\ [\,]) \\ & \to_{\mathcal{R}} & :\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0}))\ (\mathsf{map}\ (\circ\ \mathsf{s}\ \mathsf{s})\ [\,]) \\ & \to_{\mathcal{R}} & :\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0}))\ [\,]. \end{array}$$

3 Termination by reduction pairs

The *head rewrite step* $\overset{\mathrm{h}}{\to}$ is defined recursively as follows: $s \overset{\mathrm{h}}{\to} t$ if (1) $s = l\sigma$ and $t = r\sigma$ for some rewrite rule $l \to r$ and some substitution σ or (2) $s = (s_0\ u_1\ \cdots\ u_n)$, $t = (t_0\ u_1\ \cdots\ u_n)$, and $s_0 \overset{\mathrm{h}}{\to} t_0$. The *non-head rewrite step* is defined by $\overset{\mathrm{nh}}{\to} = \to \backslash \overset{\mathrm{h}}{\to}$. Let D be a set of *dependency pairs* of an STTRS $\mathcal{R}$. In simply typed term rewriting, a root rewrite step using a dependency pair is distinguished from the rewrite relation (since it is not in general type-preserving), and denoted by $\rightarrowtail_D$. A *dependency chain* of D is an infinite sequence $t_0, t_1, \ldots$ on $\mathrm{NT}_{\min}(\mathcal{R})$ such that $t_i \overset{\mathrm{nh}}{\to}{}^* \cdot \rightarrowtail_D t_{i+1}$ for all $i \geq 0$. Here, $\mathrm{NT}_{\min}(\mathcal{R})$ is the set of minimal (with respect to the subterm relation $\trianglelefteq$) non-terminating terms. The family of all minimal (with respect to the set inclusion $\subseteq$) sets of dependency pairs that admit dependency chain is denoted by $\mathbf{DC}_{\min}(\mathcal{R})$. For $D \in \mathbf{DC}_{\min}(\mathcal{R})$, every element of D occurs infinitely often in its dependency chain.

Theorem 2 (termination by reduction pairs) Let $\mathcal{R} = \langle \Sigma, R \rangle$ be an STTRS and D a finite set of dependency pairs. If there exists a reduction pair $\langle \gtrsim, > \rangle$ such that $R \subseteq \gtrsim$, $D \subseteq \gtrsim$, and $D \cap > \neq \emptyset$, then $D \notin \mathbf{DC}_{\min}$.

In contrast to the first-order case, heads of rhs of dependency pairs need not be constants in general. Based on the *head instantiation* technique [AY05], however, it suffices to handle dependency pairs whose heads of rhs are constants. Such dependency pairs are referred to as *head-instantiated dependency pairs*.

4 Argument filterings

Since argument filtering may not preserve well-typedness, we need an underlying untyped calculus. For this, the framework of *S-expression reduction systems*

(*SRSs* for short) [Toy04] is suitable. An S-expression is a first-order term with a special variadic function symbol @. The set $\mathrm{S}(\Sigma, V)$ of S-expressions is defined as: $\Sigma \cup V \subseteq \mathrm{S}(\Sigma, V)$; if $s_1, \ldots, s_n \in \mathrm{S}(\Sigma, V)$ $(n \geq 0)$ then $@(s_1, \ldots, s_n) \in \mathrm{S}(\Sigma, V)$. An S-expression $@(s_1, \ldots, s_n)$ is abbreviated as $(s_1 \ \cdots \ s_n)$. We note that () and (()(())) are also S-expressions. Each simply typed term can be regarded as an S-expression by forgetting its type information.

The first-order argument filtering is specified by function symbols, that is, $\pi(f(s_1, \ldots, s_n))$ is defined by the value of $\pi(f)$. In contrast, the head symbol of a simply typed term t is insufficient to specify filtering of t: e.g. $(f\ x\ y)$ and $((f\ x\ y)\ z)$ have the same head symbol but may have different filtering—the depth of head symbol occurrence needs to be considered additionally.

Definition 3 (filtering domain) Let X be a set of simply typed constants and simply typed variables. We define the *filtering domain* $\mathcal{D}(X) \subseteq X \times \mathbb{N}$ for X by $\mathcal{D}(X) = \bigcup_\tau \{\langle a, n\rangle \mid a \in X, a \text{ is of type } \tau, 0 \leq n < \mathrm{depth}(\tau)\}$. Here, the *depth* of a simple type τ is defined as follows: $\mathrm{depth}(\mathrm{o}) = 0$; $\mathrm{depth}(\tau_1 \times \cdots \times \tau_n \to \tau_0) = \mathrm{depth}(\tau_0) + 1$.

The *head depth* of a simply typed term t is defined as follows: $\mathrm{hdep}(a) = 0$ for $a \in \Sigma \cup V$; $\mathrm{hdep}((t_0\ t_1 \cdots t_n)) = \mathrm{hdep}(t_0) + 1$. The next lemma shows that mappings from the filtering domain are suitable to specify all argument filterings.

Lemma 4 Let X be a set of simply typed constants and simply typed variables. If s has a function type and $\mathrm{head}(s) \in X$, then $\langle \mathrm{head}(s), \mathrm{hdep}(s)\rangle \in \mathcal{D}(X)$.

The marking of head symbols similar to the first-order dependency pairs is useful to simplify the definition of argument filtering. For each $a \in \Sigma_\mathrm{d}$, let $a^\sharp$ be a new constant having the same type as a. Let $\Sigma^\sharp_\mathrm{d} = \{a^\sharp \mid a \in \Sigma_\mathrm{d}\}$ and $\Sigma^\sharp = \Sigma \cup \Sigma^\sharp_\mathrm{d}$.

For each $s \in \mathrm{T}(\Sigma, V)$ of type τ and $n \leq \mathrm{depth}(\tau)$, $\mathrm{type}(s, n)$ is defined as: $\mathrm{type}(s, 0) = \tau$; $\mathrm{type}(s, n+1) = \tau_0$ if $\mathrm{type}(s, n) = \tau_1 \times \cdots \times \tau_m \to \tau_0$. For any function type τ, $|\tau|$ is defined as: $|\tau_1 \times \cdots \times \tau_m \to \tau_0| = m$.

Definition 5 (argument filtering) Let $\mathbb{L}$ be the set of natural numbers and lists of natural numbers. An *argument filtering* π is a function from $\mathcal{D}(\Sigma^\sharp \cup V)$ to $\mathbb{L}$ such that for each $\langle f, n\rangle \in \mathcal{D}(\Sigma^\sharp \cup V)$, either $\pi(f, n) = [i_1, \ldots, i_k]$ for some $0 \leq i_1 < \cdots < i_k \leq |\mathrm{type}(f, n)|$ or $\pi(f, n) = i$ for some $0 \leq i \leq |\mathrm{type}(f, n)|$. Note that $k \geq 0$ and $k = 0$ means that the result is an empty list.

For a simply typed term t such that $\mathrm{head}(t) \in \Sigma_\mathrm{d}$, define $t^\sharp$ recursively as follows: $t^\sharp = a^\sharp$ if $t = a \in \Sigma_\mathrm{d}$; $t^\sharp = (t_0^\sharp\ t_1 \cdots t_n)$ if $t = (t_0\ t_1 \cdots t_n)$. The set of terms $\mathrm{T}(\Sigma, V) \cup \{t^\sharp \mid t \in \mathrm{T}(\Sigma, V), \mathrm{head}(t) \in \Sigma_\mathrm{d}\}$ is denoted by $\mathrm{T}^\sharp(\Sigma, V)$.

Definition 6 (application of argument filtering) Let π be an argument filtering. For each simply typed term $t \in \mathrm{T}^\sharp(\Sigma, V)$, an S-expressions $\pi(t)$ is defined as follows: (1) $\pi(a) = a$ for all $a \in \Sigma^\sharp \cup V$; (2) $\pi((t_0\ t_1 \cdots t_n)) = (\pi(t_{i_1})\ \cdots\ \pi(t_{i_k}))$ if $\pi(\mathrm{head}(t_0), \mathrm{hdep}(t_0)) = [i_1, \ldots, i_k]$; (3) $\pi((t_0\ t_1 \cdots t_n)) = \pi(t_i)$ if $\pi(\mathrm{head}(t_0), \mathrm{hdep}(t_0)) = i$.

Filtering functions should consistently select the same argument positions from both a term with head variable and its instance.

Example 7 (unsound filtering (1)) Let $\mathcal{R} = \langle \Sigma, R \rangle$ be an STTRS where $\Sigma = \{\ \mathsf{0}^{\mathsf{o}},\ \mathsf{f}^{\mathsf{o}\to\mathsf{o}},\ \mathsf{s}^{\mathsf{o}\to\mathsf{o}}\ \}$ and

$$R = \left\{\ \mathsf{f}\ (F\ x) \quad\to\quad \mathsf{f}\ (\mathsf{s}\ x)\ \right\}.$$

If π is an argument filtering such that $\pi(\mathsf{s}, 0) = 1$ and $\pi(\mathsf{f}, 0) = \pi(\mathsf{f}^\sharp, 0) = \pi(F, 0) = [0, 1]$, the following satisfiable set of constraints is obtained: $\{\mathsf{f}\ (F\ x) \geq \mathsf{f}\ x,\ \mathsf{f}^\sharp\ (F\ x) > \mathsf{f}^\sharp\ x\}$. Since $\mathcal{R}$ is not terminating, this argument filtering is unsound.

Filtering functions should consistently select the same argument positions from a term when its head is rewritten by a rule of function type.

Example 8 (unsound filtering (2)) Let $\mathcal{R} = \langle \Sigma, R \rangle$ be an STTRS where $\Sigma = \{\ \mathsf{f}^{\mathsf{o}\to\mathsf{o}},\ \mathsf{g}^{\mathsf{o}\to\mathsf{o}},\ \mathsf{h}^{\mathsf{o}\to\mathsf{o}}\ \}$ and

$$R = \left\{ \begin{array}{lcl} \mathsf{f}\ (\mathsf{h}\ x) & \to & \mathsf{f}\ (\mathsf{g}\ x) \\ \mathsf{g} & \to & \mathsf{h} \end{array} \right\}.$$

Let $D = \{\mathsf{f}^\sharp\ (\mathsf{h}\ x) \rightarrowtail \mathsf{f}^\sharp\ (\mathsf{g}\ x)\}$. If π is an argument filtering such that $\pi(\mathsf{f}, 0) = \pi(\mathsf{g}, 0) = [\,]$, $\pi(\mathsf{f}^\sharp, 0) = [0, 1]$, and $\pi(\mathsf{h}, 0) = [1]$, the following satisfiable set of constraints is obtained: $\{() \geq (), \mathsf{g} \geq \mathsf{h}, (\mathsf{f}^\sharp\ (x)) > (\mathsf{f}^\sharp\ ())\}$. Since $\mathcal{R}$ is not terminating, this argument filtering is unsound.

Definition 9 (stabilization domain) Let π be an argument filtering. For any simply typed term $t \in \mathrm{T}^\sharp(\Sigma, V)$, the set $\mathrm{SDom}(t) \subseteq \mathcal{D}(\Sigma \cup V)$ of *stabilization domain* of t is defined as: $\mathrm{SDom}(a) = \emptyset$; $\mathrm{SDom}((t_0\ t_1 \cdots t_n)) = \bigcup\{\mathrm{SDom}(t_{i_j}) \mid 1 \leq j \leq k\}$ if $\pi(\mathrm{head}(t_0), \mathrm{hdep}(t_0)) = [i_1, \ldots, i_k]$ and $\mathrm{head}(t_0) \in \Sigma^\sharp_{\mathrm{d}}$; $\mathrm{SDom}((t_0\ t_1 \cdots t_n)) = \{\langle \mathrm{head}(t_0), \mathrm{hdep}(t_0)\rangle\} \cup \bigcup\{\mathrm{SDom}(t_{i_j}) \mid 1 \leq j \leq k\}$ if $\pi(\mathrm{head}(t_0), \mathrm{hdep}(t_0)) = [i_1, \ldots, i_k]$ and $\mathrm{head}(t_0) \in \Sigma \cup V$; $\mathrm{SDom}((t_0\ t_1 \cdots t_n)) = \mathrm{SDom}(t_i)$ if $\pi(\mathrm{head}(t_0), \mathrm{hdep}(t_0)) = i$ and $\mathrm{head}(t_0) \in \Sigma^\sharp_{\mathrm{d}}$; $\mathrm{SDom}((t_0\ t_1 \cdots t_n)) = \{\langle \mathrm{head}(t_0), \mathrm{hdep}(t_0)\rangle\} \cup \mathrm{SDom}(t_i)$ if $\pi(\mathrm{head}(t_0), \mathrm{hdep}(t_0)) = i$ and $\mathrm{head}(t_0) \in \Sigma \cup V$.

Definition 10 (stability) Let X be a set of simply typed constants and simply typed variables. Let f be a function from $\mathcal{D}(X)$ to $\mathbb{L}$. (1) f is stable w.r.t. a simple type τ if for any $\langle a, n\rangle, \langle b, m\rangle \in \mathcal{D}(X)$, $\mathrm{type}(a, n) = \mathrm{type}(b, m) = \tau$ implies $f(a, n) = f(b, m)$. (2) f is stable w.r.t. $A \subseteq \mathcal{D}(X)$ if f is stable w.r.t. any τ in the set $\{\mathrm{type}(a, n) \mid \langle a, n\rangle \in A, a \in V\}$.

Definition 11 (stability w.r.t. rules) Let π be an argument filtering and $\pi^{\mathrm{nh}} = \pi{\downarrow}\mathcal{D}(\Sigma, V)$. Here, $\downarrow$ denotes the operation of restricting the domain.

1. π is *stable* w.r.t. a set R of simply typed rewrite rules if π^{nh} is stable w.r.t. $\bigcup_{l\to r\in R} \mathrm{SDom}(l) \cup \mathrm{SDom}(r)$ and if R contains a simply typed rewrite rule of function type τ then π^{nh} is stable w.r.t. $\tau, \ldots, \tau + (\mathrm{depth}(\tau) - 1)$.

2. π is *stable* w.r.t. a set D of simply typed dependency pairs if π^{nh} is stable w.r.t. $\bigcup_{l\to r\in D} \mathrm{SDom}(l^\sharp) \cup \mathrm{SDom}(r^\sharp)$.

Theorem 12 (argument filtering refinement) Let $\mathcal{R} = \langle \Sigma, R \rangle$ be an STTRS and D a finite set of head-instantiated dependency pairs. If there exists a reduction pair $\langle \gtrsim, > \rangle$ on $\mathrm{S}(\Sigma, V)$ and an argument filtering π stable w.r.t. R and D, $\pi(R) \subseteq \gtrsim$, $\pi(D^\sharp) \subseteq \gtrsim$, and $\pi(D^\sharp) \cap > \neq \emptyset$, then $D \notin \mathbf{DC}_{\min}$.

5 Usable rules

Definition 13 (usable rules) We write $f \blacktriangleright g$ when there exists a simply typed rewrite rule $l \to r \in R$ such that $\mathrm{head}(l) = f$ and $g \in \Sigma_{\mathrm{d}}(r)$. We denote the reflexive transitive closure of $\blacktriangleright$ by $\blacktriangleright^*$. For a set D of head-instantiated dependency pairs,

$$\mathcal{U}_R(D^\sharp) = \{l \to r \in R \mid f \blacktriangleright^* \mathrm{head}(l) \text{ for some } f \in \Sigma_{\mathrm{d}}(\mathrm{RHS}(D^\sharp))\}.$$

Let $\mathcal{C}_\mathcal{E} = \langle \{\mathsf{nil}, \mathsf{cons}\}, C_\mathcal{E} \rangle$ be an SRS where $C_\mathcal{E} = \{(\mathsf{cons}\ x\ y) \to x, (\mathsf{cons}\ x\ y) \to y\}$. Let us first explain that a naive extension of usual first-order usable rules criteria is not adapted to the higher-order setting.

Example 14 (counterexample) Let $\mathcal{R} = \langle \Sigma, R \rangle$ be an STTRS where $\Sigma = \{ \mathsf{0}^{\mathrm{o}}, \mathsf{f}^{(\mathrm{o}\to\mathrm{o})\times\mathrm{o}\to\mathrm{o}}, \mathsf{g}^{\mathrm{o}\to\mathrm{o}} \}$ and

$$R = \left\{ \begin{array}{lcl} \mathsf{f}\ F\ \mathsf{0} & \to & \mathsf{f}\ F\ (F\ \mathsf{0}) \\ \mathsf{g}\ \mathsf{0} & \to & \mathsf{0} \end{array} \right\}.$$

For $D = \{\mathsf{f}\ F\ \mathsf{0} \rightarrowtail \mathsf{f}\ F\ (F\ \mathsf{0})\}$, there is an infinite dependency chain $\mathsf{f}\ \mathsf{g}\ \mathsf{0} \rightarrowtail_D \mathsf{f}\ \mathsf{g}\ (\mathsf{g}\ \mathsf{0}) \to_{\mathcal{R}} \mathsf{f}\ \mathsf{g}\ \mathsf{0} \rightarrowtail_D \cdots$. However $\mathcal{U}_R(D^\sharp) = \emptyset$ and thus there is no infinite dependency chain on D and $C_\mathcal{E} \cup \mathcal{U}_R(D^\sharp)$.

Definition 15 (higher-order usable rules)

1. The *range-order* $\trianglerighteq$ is the smallest partial order on ST satisfying $\tau_1 \times \cdots \times \tau_n \to \tau_0 \trianglerighteq \tau_0$.
2. We write $f \blacktriangleright_{\mathrm{h}} g$ when (1) there exists a simply typed rewrite rule $l \to r \in R$ such that $\mathrm{head}(l) = f$ and $g \in \Sigma_{\mathrm{d}}(r)$, or (2) there exists a simply typed rewrite rule $l \to r \in R$ and $F^\tau \in \mathrm{PV}(r)$ such that $\mathrm{head}(l) = f$, $g^\rho \in \Sigma_{\mathrm{d}}$ for some $\rho \trianglerighteq \tau$. We denote the reflexive transitive closure of $\blacktriangleright_{\mathrm{h}}$ by $\blacktriangleright_{\mathrm{h}}^*$.
3. Let D be a set of head-instantiated dependency pairs. Then

$$\mathcal{U}^{\mathrm{h}}_R(D^\sharp) = \{l \to r \in R \mid f \blacktriangleright_{\mathrm{h}}^* \mathrm{head}(l) \text{ for some } f \in \Sigma_{\mathrm{d}}(\mathrm{RHS}(D^\sharp))\}$$

Theorem 16 (usable rules refinement) Let $\mathcal{R} = \langle \Sigma, R \rangle$ be an STTRS and D a finite set of head-instantiated dependency pairs. If there exists a reduction pair $\langle \gtrsim, > \rangle$ on $\mathrm{S}(\Sigma^\sharp \cup \{\mathsf{cons}, \mathsf{nil}\}, V)$ such that $C_\mathcal{E} \cup \mathcal{U}^{\mathrm{h}}_R(D^\sharp) \subseteq \gtrsim$, $D^\sharp \subseteq \gtrsim$, and $D^\sharp \cap > \neq \emptyset$, then $D \notin \mathbf{DC}_{\min}$.

Example 17 (termination proof) Let $\mathcal{R}$ be the STTRS of Example 1. The set of dependency pairs of $\mathcal{R}$ is as follows:

$$\left\{ \begin{array}{lllll} (5) & \mathsf{map}\ F\ (:\ x\ xs) & \rightarrowtail & F\ x \\ (6) & \mathsf{map}\ F\ (:\ x\ xs) & \rightarrowtail & \mathsf{map}\ F\ xs \\ (7) & (\circ\ F\ G)\ x & \rightarrowtail & F\ (G\ x) \\ (8) & (\circ\ F\ G)\ x & \rightarrowtail & G\ x \\ (9) & \mathsf{twice}\ F & \rightarrowtail & \circ\ F\ F \\ (10) & \mathsf{twice}\ F & \rightarrowtail & \circ \\ (11) & (\mathsf{twice}\ F)\ x & \rightarrowtail & (\circ\ F\ F)\ x \end{array} \right\}.$$

Two SCCs are obtained from its (approximated) dependency graph—namely, $\{(6)\}$ and $\{(7), (8), (9)\}$. Let $D = \{(7), (8), (9)\}$. The head instantiation and

head marking of D yields the following set D' of simply typed dependency pairs:

$$\left\{\begin{array}{lll}
(7a) & (\circ^\sharp\ (\circ\ U\ V)\ G)\ x & \rightarrowtail\ (\circ^\sharp\ U\ V)\ (G\ x) \\
(7b) & (\circ^\sharp\ (\mathsf{twice}\ U)\ G)\ x & \rightarrowtail\ (\mathsf{twice}^\sharp\ U)\ (G\ x) \\
(8a) & (\circ^\sharp\ F\ (\circ\ U\ V))\ x & \rightarrowtail\ (\circ^\sharp\ U\ V)\ x \\
(8b) & (\circ^\sharp\ F\ (\mathsf{twice}\ U))\ x & \rightarrowtail\ (\mathsf{twice}^\sharp\ U)\ x \\
(9) & (\mathsf{twice}^\sharp\ F)\ x & \rightarrowtail\ (\circ^\sharp\ F\ F)\ x
\end{array}\right\}.$$

We have $\mathcal{U}^{\mathrm{h}}_R(D') = \{(3), (4)\}$. By taking a stable argument filtering π such that $\pi(F^{\mathrm{o}\to\mathrm{o}}, 0) = \pi(\circ, 1) = \pi(\mathsf{twice}, 1) = 1$, $\pi(\mathsf{twice}, 0) = \pi(\mathsf{twice}^\sharp, 0) = \pi(\circ^\sharp, 1) = [0, 1]$, and $\pi(\circ, 0) = \pi(\circ^\sharp, 0) = [0, 1, 2]$, we get the following set of constraints:

$$\left\{\begin{array}{lcl}
x & \geq & x \\
\mathsf{twice}\ F & \geq & \circ\ F\ F \\
(\circ^\sharp\ (\circ\ U\ V)\ G)\ x & > & (\circ^\sharp\ U\ V)\ (G\ x) \\
(\circ^\sharp\ (\mathsf{twice}\ U)\ G)\ x & > & (\mathsf{twice}^\sharp\ U)\ (G\ x) \\
(\circ^\sharp\ F\ (\circ\ U\ V))\ x & > & (\circ^\sharp\ U\ V)\ x \\
(\circ^\sharp\ F\ (\mathsf{twice}\ U))\ x & > & (\mathsf{twice}^\sharp\ U)\ x \\
(\mathsf{twice}^\sharp\ F)\ x & > & (\circ^\sharp\ F\ F)\ x
\end{array}\right\}.$$

All constraints are satisfied by the lexicographic path ordering for S-expressions [Toy04] with the precedence $\mathsf{twice} > \mathsf{twice}^\sharp > \circ^\sharp$ and $\mathsf{twice} > \circ > \circ^\sharp$. It is not hard to show $\{(6)\} \notin \mathbf{DC}_{\min}$ in a similar way. Thus $\mathcal{R}$ is terminating.

References

[AG00] T. Arts and J. Giesl. Termination of term rewriting using dependency pairs. *TCS*, 236(1–2):133–178, 2000.

[AY05] T. Aoto and T. Yamada. Dependency pairs for simply typed term rewriting. In *Proc. of RTA 2005*, volume 3467 of *LNCS*, pages 120–134. Springer-Verlag, 2005.

[Bla06] F. Blanqui. Higher-order dependency pairs. In *Proc. of WST 2006*, pages 22–26, 2006.

[GTSK05] J. Giesl, R. Thiemann, and P. Schneider-Kamp. Proving and disproving termination of higher-order functions. In *Proc. of FroCoS 2005*, volume 3717 of *LNAI*, pages 216–231. Springer-Verlag, 2005.

[HM04] N. Hirokawa and A. Middeldorp. Dependency pairs revisited. In *Proc. of RTA 2004*, volume 3091 of *LNCS*, pages 249–268. Springer-Verlag, 2004.

[SK05] M. Sakai and K. Kusakari. On dependency pair method for proving termination of higher-order rewrite systems. *IEICE Trans. on Inf. & Sys.*, E88-D(3):583–593, 2005.

[TGSK04] R. Thiemann, J. Giesl, and P. Schneider-Kamp. Improved modular termination proofs using dependency pairs. In *Proc. of IJCAR 2004*, volume 3097 of *LNAI*, pages 75–90. Springer-Verlag, 2004.

[Toy04] Y. Toyama. Termination of S-expression rewriting systems: Lexicographic path ordering for higher-order terms. In *Proc. of RTA 2004*, volume 3091 of *LNCS*, pages 40–54. Springer-Verlag, 2004.

[Yam01] T. Yamada. Confluence and termination of simply typed term rewriting systems. In *Proc. of RTA 2001*, volume 2051 of *LNCS*, pages 338–352. Springer-Verlag, 2001.