Decidability of Termination for Semi-Constructor TRSs, Left-Linear Shallow TRSs and Related Systems

Yi Wang* and Masahiko Sakai
Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8603 Japan
{ywang80@trs.cm.,sakai@}is.nagoya-u.ac.jp

September 12, 2006

Abstract

We consider several classes of term rewriting systems and prove that termination is decidable for these classes. By showing the cycling property of infinite dependency chains, we prove that termination is decidable for semi-constructor case, which is a superclass of right-ground TRSs. By analyzing argument propagation cycles in the dependency graph, we show that termination is also decidable for left-linear shallow TRSs. Moreover we extend these by combining these two techniques.

1 Introduction

Termination is one of the central properties of term rewriting systems (TRSs for short). We say a TRS terminates if it does not admit any infinite reduction sequences. Termination guarantees that any expression cannot be infinitely rewritten, and hence the existence of a normal form for it. As we go from simple to more general classes of term rewriting systems, the difficulty of deciding termination increases until it becomes undecidable. It is meaningful to identify the decidability barrier and study decidability issues for some intermediate classes, especially if these classes are expressive enough to capture interesting rules.

As a generalization of the decidable classes of ground TRSs [7] and right-ground TRSs [4], the class of semi-constructor TRSs is studied. A TRS is called semi-constructor if every defined symbol in the right-hand sides of rules takes ground terms as its arguments. By showing the cycling property of infinite dependency chains, we give a positive answer to this problem.

The class of shallow TRSs has been attracting some interests from researchers due to the decidability of reachability and joinability problems for this class [3, 8, 10, 13]. A TRS is called shallow if all variables in l, r occur at positions with depth 0 or 1 for each rule $l \to r$. In 2005, the affirmative result on termination of TRSs that contains right-linear shallow rules was shown by Godoy and Tiwari [5]. Here we propose a technique based on the analysis of argument propagation in the dependency graph.

Combining the two techniques for semi-constructor case and shallow case, we prove the decidability of termination for the following TRSs:

1. right-linear reverse-growing TRSs with all the dependency pairs being shallow or right-ground
2. left-linear growing TRSs with all the dependency pairs being shallow or right-ground.

The organization of this paper is as follows. In Section 2, we review preliminary definitions of term rewriting systems and introduce basic definitions and results concerning dependency pair method that will be used in Section 3. In Section 3, we give the definition of loop, head-loop and cycle first, then list our results and give their proofs. In Section 4, we compare our results with some existing results.

*Presently, with Financial Services Dept., Accenture Japan Ltd.

2 Preliminaries

We assume the reader is familiar with the standard definitions of term rewriting systems [2] and here we just review the main notations used in this paper.

A *signature* $\mathcal{F}$ is a set of function symbols, where every $f \in \mathcal{F}$ is associated with a non-negative integer by an arity function: $arity\colon \mathcal{F} \to \mathbb{N}(=\{0,1,2,\ldots\})$. Function symbols of arity 0 are called *constants*. The set of all *terms* built from a signature $\mathcal{F}$ and a countable infinite set $\mathcal{V}$ of *variables* such that $\mathcal{F} \cap \mathcal{V} = \emptyset$, is represented by $\mathcal{T}(\mathcal{F},\mathcal{V})$. The set of *ground terms* is denoted by $\mathcal{T}(\mathcal{F},\emptyset)$ ($\mathcal{T}(\mathcal{F})$ for short). We write $s = t$ when two terms s and t are identical. The *root symbol* of a term t is denoted by $root(t)$.

The set of all *positions* in a term t is denoted by $\mathcal{P}os(t)$ and ε represents the root position. We denote the *subterm ordering* by $\trianglelefteq$, that is, $t \trianglelefteq s$ if t is a subterm of s, and $t \triangleleft s$ if $t \trianglelefteq s$ and $t \neq s$. The *depth* of a position $p \in \mathcal{P}os(t)$ is $|p|$. The *height* of a term t is 0 if t is a variable or a constant, and $1 + max(\{height(s_i) \mid i \in \{1,\ldots,m\}\})$ if $t = f(s_1,\ldots,s_m)$. Let C be a *context* with a hole $\Box$. We write $C[t]$ for the term obtained from C by replacing $\Box$ with a term t.

A *substitution* θ is a mapping from $\mathcal{V}$ to $\mathcal{T}(\mathcal{F},\mathcal{V})$ such that the set $\mathrm{Dom}(\theta) = \{x \in \mathcal{V} \mid \theta(x) \neq x\}$ is finite. We usually identify a substitution θ with the set $\{x \mapsto \theta(x) \mid x \in \mathrm{Dom}(\theta)\}$ of variable bindings. In the following, we write $t\theta$ instead of $\theta(t)$.

A *rewrite rule* $l \to r$ is a directed equation which satisfies $l \notin \mathcal{V}$ and $\mathrm{Var}(r) \subseteq \mathrm{Var}(l)$. A *term rewriting system* TRS is a finite set of rewrite rules. If the two conditions $l \notin \mathcal{V}$ and $\mathrm{Var}(r) \subseteq \mathrm{Var}(l)$ are not imposed, then we call it *eTRS*. We use R^{-1} for the reverse eTRS of R; $R^{-1} = \{r \to l \mid l \to r \in R\}$. The *reduction relation* $\to_R \subseteq \mathcal{T}(\mathcal{F},\mathcal{V}) \times \mathcal{T}(\mathcal{F},\mathcal{V})$ associated with a TRS R is defined as follows: $s \to_R t$ if there exist a rewrite rule $l \to r \in R$, a substitution θ, and a context C such that $s = C[l\theta]$ and $t = C[r\theta]$. The subterm $l\theta$ of s is called a *redex* and we say that s is reduced to t by contracting redex $l\theta$. The transitive closure of $\to_R$ is denoted by $\to_R^+$. The transitive and reflexive closure of $\to_R$ is denoted by $\to_R^*$. If $s \to_R^* t$, then we say that there is a *reduction sequence* starting from s to t or t is *reachable* from s by R. A term without redexes is called a *normal form*. A rewrite rule $l \to r$ is called *left-linear* (resp. *right-linear*) if no variable occurs twice in l (resp. r). It is called *linear* if it is both left- and right-linear. A TRS is called left-linear (resp. right-linear, resp. linear) if all of its rules are left-linear (resp. right-linear, resp. linear).

For a TRS R, a term $t \in \mathcal{T}(\mathcal{F},\mathcal{V})$ *terminates* if there is no infinite reduction sequence starting from t. We say that R terminates if every term terminates.

For a TRS R, a function symbol $f \in \mathcal{F}$ is a *defined symbol* of R if $f = root(l)$ for some rule $l \to r \in R$. The set of all defined symbols of R is denoted by $D_R = \{root(l) \mid \exists l \to r \in R\}$. We write C_R for the set of all *constructor symbols* of R which is defined as $\mathcal{F} \backslash D_R$. A term t has a *defined root symbol* if $root(t) \in D_R$.

Let R be a TRS over a signature $\mathcal{F}$. $\mathcal{F}^\sharp$ denotes the union of $\mathcal{F}$ and $D_R^\sharp = \{f^\sharp \mid f \in D_R\}$ where $\mathcal{F} \cap D_R^\sharp = \emptyset$ and $f^\sharp$ has the same arity as f. We call these fresh symbols *dependency pair symbols*. Given a term $t = f(t_1,\ldots,t_n) \in \mathcal{T}(\mathcal{F},\mathcal{V})$ with f defined, we write $t^\sharp$ for the term $f^\sharp(t_1,\ldots,t_n)$. If $l \to r \in R$ and u is a subterm of r with a defined root symbol, then the rewrite rule $l^\sharp \to u^\sharp$ is called a *dependency pair* of R. The set of all dependency pairs of R is denoted by $\mathrm{DP}(R)$.

For TRSs R and $\mathcal{C}$, a (possibly infinite) sequence of the elements of $\mathcal{C}$ $s_1^\sharp \to t_1^\sharp$, $s_2^\sharp \to t_2^\sharp$, ... is an $(R,\mathcal{C})$-*chain* if there exist substitutions $\tau_1, \tau_2, \ldots$ such that $t_i^\sharp\tau_i \to_R^* s_{i+1}^\sharp\tau_{i+1}$ holds for every $s_i^\sharp \to t_i^\sharp$ and $s_{i+1}^\sharp \to t_{i+1}^\sharp$ in the sequence. An $(R,\mathrm{DP}(R))$-chain is called *dependency chain*.

Theorem 1 ([1, 6]) For a TRS R, R does not terminate if and only if there exists an infinite dependency chain.

The nodes of an $(R,\mathcal{C})$-*graph* denoted by $\mathrm{G}(R,\mathcal{C})$ are the elements of $\mathcal{C}$ and there is an edge from a node $s^\sharp \to t^\sharp$ to $u^\sharp \to v^\sharp$ if and only if there exist substitutions σ and τ such that $t^\sharp\sigma \to_R^* u^\sharp\tau$. An $(R,\mathrm{DP}(R))$-graph is called *dependency graph* and denoted by $\mathrm{DG}(R)$. Note that the dependency graph is not computable in general. However, our results will work on any approximation of the dependency graph. We say a graph is an *approximate graph* of a $(R,\mathcal{C})$-graph G if it contains G as a subgraph and $root(t) = root(u)$ for each arrow from a node $s^\sharp \to t^\sharp$ to $u^\sharp \to v^\sharp$. We remark that there exists at least one computable approximate graph for every $(R,\mathcal{C})$-graph.

The special notations "$(R,\mathcal{C})$-chain" and "$(R,\mathcal{C})$-graph" adopted in this paper is for handling left-linear TRSs as right-linear ones. For example, we will use an "$(R^{-1},\mathrm{DP}(R)^{-1})$-chain".

3 Decidability of Termination Based on Cycle Detection

Infinite reduction sequences are often composed of cycles. A cycle is a reduction sequence where a term is rewritten to the same term. More generally, a loop is a reduction sequence where an instance of the starting term re-occurs as a subterm. It is obvious that a loop gives an infinite reduction sequence. In fact, the usual way to deduce non-termination is to construct a loop.

Definition 2 (Loop, Head-Loop, Cycle)

1. A *reduction sequence loops* if it contains $t' \to_R^+ C[t'\theta]$ for some context C, substitution θ and term t'. Similarly, a reduction sequence *head-loops* if containing $t' \to_R^+ t'\theta$, and *cycles* if containing $t' \to_R^+ t'$.
2. A *term t loops* (resp. *head-loops*, resp. *cycles*) with respect to R if there is a looping (resp. head-looping, resp. cycling) reduction sequence starting from t.
3. A *TRS R admits a loop* (resp. *head-loop*, resp. *cycle*) if there is a term t such that t loops (resp. head-loops, resp. cycles) with respect to R.

Proposition 3 The following statements hold:

1. If t cycles, then t head-loops. If t head-loops, then t loops.
2. A TRS does not terminate if it admits a loop or a head-loop or a cycle.

Example 4 Let $R_1 = \{f(x) \to h(f(g(a))), g(x) \to g(h(x))\}$ and $t = f(x)$. We can construct the following reduction sequence by only applying the former rule: $f(x) \to h(f(g(a))) \to h(h(f(g(a)))) \to \cdots$ which loops with $C = h[\Box]$, $\theta = \{x \mapsto g(a)\}$ and $t' = f(x)$. Notice there are more than one looping reduction sequences for R_1.

Naturally, the observation above inspires us to find some class of TRSs, whose non-termination is *equivalent to* the existence of loops. If we are able to check the existence of loops, then termination of such a class becomes decidable.

The following theorem lists our main results and will be proved in the following subsections.

Theorem 5 Termination of the following classes of TRSs is decidable:

1. semi-constructor TRSs
2. right-linear shallow TRSs
3. left-linear shallow TRSs
4. right-linear rev-growing TRSs with all the dependency pairs being shallow or right-ground.
5. left-linear growing TRSs with all the dependency pairs being shallow or right-ground.

3.1 Semi-Constructor TRSs

Definition 6 (Semi-Constructor TRS) A term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ is a *semi-constructor* term if every term s such that $s \trianglelefteq t$ and $root(s) \in D_R$ is ground. A TRS R is a semi-constructor system if r is a semi-constructor term for every rule $l \to r \in R$.

Example 7 The TRS $R_2 = \{f(x) \to h(x, f(g(a))), g(x) \to g(h(a, a))\}$ is a semi-constructor system.

Proposition 8 A TRS R is called *right-ground* if for every $l \to r \in R$, r is ground. The following statements hold:

1. Right-ground TRSs are semi-constructor systems.
2. For a semi-constructor TRS R, rules in $\mathrm{DP}(R)$ are right-ground.

For a given TRS, let $\mathcal{T}_\infty$ denote the set of all *minimal non-terminating terms*, here "minimal" is used in the sense that all its proper subterms terminate.

Definition 9 ($\mathcal{C}$-min) For a TRS R, let $\mathcal{C} \subseteq \mathrm{DP}(R)$. An infinite reduction sequence in $R \cup \mathcal{C}$ in the form $t_1^\sharp \to_R^* t_2^\sharp \to_{\mathcal{C}} t_3^\sharp \to_R^* t_4^\sharp \to_{\mathcal{C}} \cdots$ with $t_i \in \mathcal{T}_\infty$ for all $i \geq 1$ is called a *$\mathcal{C}$-min reduction sequence*. We use $\mathcal{C}_{min}(t^\sharp)$ to denote the set of all $\mathcal{C}$-min reduction sequences starting from $t^\sharp$.

Proposition 10 ([1, 6]) Given a TRS R, we have the following statements:

1. If there exists an infinite dependency chain, then $\mathcal{C}_{min}(t^\sharp) \neq \emptyset$ for some $\mathcal{C} \subseteq \mathrm{DP}(R)$ and $t \in \mathcal{T}_\infty$.
2. For any sequence in $\mathcal{C}_{min}(t^\sharp)$, reduction $\to_R$ takes place below the root while reduction $\to_{\mathcal{C}}$ takes place at the root.
3. For any sequence in $\mathcal{C}_{min}(t^\sharp)$, subsequence $s^\sharp \to_{R\cup\mathcal{C}}^* t^\sharp$ implies $s \to_R^* C[t]$ for some context C.
4. For any sequence in $\mathcal{C}_{min}(t^\sharp)$, there is at least one rule in $\mathcal{C}$ which is applied infinitely often.

Lemma 11 For a TRS R, if $sq \in \mathcal{C}_{min}(t^\sharp)$ loops, then sq head-loops.

Proof: Let $sq \in \mathcal{C}_{min}(t^\sharp)$ loops, then there is a subsequence $t_k^\sharp \to_{R\cup\mathcal{C}}^+ C[t_k^\sharp\theta]$ in sq. From Prop.10–(2) and the fact that dependency pair symbols appears only in dependency pairs, we have $C[t_k^\sharp\theta] = t_k^\sharp\theta$, which implies that sq head-loops. □

Lemma 12 For a TRS R, if $sq \in \mathcal{C}_{min}(t^\sharp)$ loops, then there is a term $t_k^\sharp$ in sq such that t_k loops with respect to R.

Proof: From Lemma 11 and Prop. 10–(3). □

Lemma 13 For a semi-constructor TRS R, the following statements are equivalent:

1. R does not terminate.
2. There exists $l^\sharp \to u^\sharp \in \mathrm{DP}(R)$ such that sq cycles for some $sq \in \mathcal{C}_{min}(u^\sharp)$.

Proof: (2⇒1): It is obvious by Lemma 12. (1⇒2): By Theorem 1, there exists an infinite dependency chain. By Prop. 10–(1), there exists a sequence $sq \in \mathcal{C}_{min}(t^\sharp)$. By Prop. 10–(4), there is some rule $l^\sharp \to u^\sharp \in \mathcal{C}$ which is applied at root reduction in sq infinitely often. By Prop. 8–(2), $u^\sharp$ is ground. Thus $u^\sharp$ cycles in the form $u^\sharp \to_{R\cup DP(R)}^* \cdot \to_{\{l^\sharp\to u^\sharp\}} u^\sharp$ in sq. □

Notice that non-termination of semi-constructor systems depends on the existence of a *cycling dependency chain*, which represents the cycle "$u^\sharp \to_{R\cup DP(R)}^* \cdot \to_{\{l^\sharp\to u^\sharp\}} u^\sharp$ in sq" in the proof of Lemma 13. Here, *cycle* is guaranteed by the fact that $\mathrm{DP}(R)$ is right-ground.

Proof: **(Theorem 5–(1))** The decision procedure for termination of semi-constructor TRS R is as follows: consider all terms $u_1, u_2, \ldots, u_n$ corresponding to the right-hand sides of $\mathrm{DP}(R) = \{l_i^\sharp \to u_i^\sharp \mid 1 \leq i \leq n\}$, and simultaneously generate all reduction sequences with respect to R starting from $u_1, u_2, \ldots, u_n$. It terminates if it enumerates all reachable terms exhaustively or it detects a looping reduction sequence $u_i \to_R^+ C[u_i]$ for some i.

Suppose R does not terminate. By Lemma 12, 13 and the groundness of u_i's, we have a looping reduction sequence $u_i \to_R^+ C[u_i]$ for some i and C. Hence we detect non-termination of R. If R terminates, then the execution of the reduction sequence generation stops finally since it is finitely branching. Thus we detect termination of R after finitely many steps. □

Next we make a natural extension by relaxing the condition for assuring *cycling*, which is mainly used in the Subsection 3.3.

Lemma 14 Let R be a TRS whose termination is equivalent to the non-existence of a dependency chain that contains infinite use of right-ground dependency pairs. Then termination of R is decidable.

Proof: We apply the above procedure starting from terms $u_1, u_2, \ldots, u_n$, where $u_i^\sharp$'s are all ground right-hand sides of dependency pairs. Suppose R is non-terminating, we have a dependency chain with infinite use of a right-ground dependency pair. Similarly to the semi-constructor case, we have a loop $u_i \to_R^+ C[u_i]$, which can be detected by the procedure. □

Example 15 Let $R_3 = \{f(a) \to g(b),\, g(x) \to f(x),\, h(a,x) \to h(b,x)\}$. We can compute the dependency graph. It has only one cycle, which contains a right ground node. From Lemma 14 we can show termination of R_3 by the procedure starting from $g(b)$.

3.2 Right-Linear Shallow or Left-Linear Shallow TRSs

In this subsection, we show how to analyze cycle of dependency chains that consist only of right-linear shallow dependency pairs and then show the decidability of termination for right-linear shallow TRSs and left-linear shallow TRSs.

Definition 16 (Shallow TRS) A rewrite rule $l \to r$ is *shallow* if all variables in $\mathrm{Var}(l) \cup \mathrm{Var}(r)$ occur at positions with depth 0 or 1. An eTRS is *shallow* if all its rewrite rules are shallow.

Example 17 TRS $R_4 = \{\, f(x,y) \to f(g(a),y),\, f(g(a),z) \to f(z,b) \,\}$ and $R_5 = \{\, g(x,x) \to f(x,a),\, f(c,x) \to g(x,b),\, a \to c,\, b \to c \,\}$ are shallow.

We say that T *is joinable to* s if $\forall t \in T.\ t \to_R^* s$ and T *is joinable* if it is joinable to some s. From now on, we assume R in which both of the following properties are decidable.

Ground Reachability: $t \to_R^* s$ for given ground terms t and s.

Ground Joinability: T is joinable for a given set T of ground terms.

For dependency chains composed of shallow dependency pairs, all informations carried by variables are passed to the next dependency pairs in its derivation. For example, consider R_5 and an infinite sequence of dependency pairs:

$$g^\sharp(\underline{x},\underline{x}) \to f^\sharp(\underline{x},a),\ f^\sharp(c,\underline{x}) \to g^\sharp(\underline{x},b),\ g^\sharp(\underline{x},\underline{x}) \to f^\sharp(\underline{x},a),\ \ldots\ .$$

It is a dependency chain because we have a derivation:

$$g^\sharp(\underline{c},\underline{c}) \to f^\sharp(\underline{c},a) = f^\sharp(c,\underline{a}) \to g^\sharp(\underline{a},b) \to \cdot \to g^\sharp(\underline{c},\underline{c}) \to f^\sharp(\underline{c},a) \to \cdots\ .$$

In order to analyze such information flows caused by variables, we introduce some notions. We refer the set of variables appears in a rule $l \to r$ as $Var(l \to r)$. We represent mappings $\Delta : \mathcal{V} \to \mathcal{P}(\mathcal{T}(\mathcal{F}))$ as $\{x \mapsto \Delta(x) \mid x \in \mathrm{Dom}(\Delta)\}$ in similar to substitutions.

Definition 18 (Labeling Function) Let R and $\mathcal{C}$ be eTRSs and AG be an approximate graph of $(R,\mathcal{C})$-graph. Let p be a path $nd_1, nd_2, nd_3 \cdots$ in AG. A labeling function L_p that associate each positive integer i with a pair of a node nd_i and a mapping $\Delta : Var(nd_i) \to \mathcal{P}(\mathcal{T}(\mathcal{F}))$ is defined as follows:

1. Let nd_1 be $f^\sharp(t_1,\ldots,t_n) \to g^\sharp(s_1,\ldots,s_m)$. Then $L_p(1) = (nd_1, \Delta_1)$ where $\Delta_1(x) = \emptyset$ for $x \in Var(nd_1)$.
2. Let $nd_i = h^\sharp(v_1,\ldots,v_k) \to f^\sharp(u_1,\ldots,u_n)$, $L_p(i) = (nd_i, \Delta_i)$ and $nd_{i+1} = f^\sharp(t_1,\ldots,t_n) \to g^\sharp(s_1,\ldots,s_m)$. Then $L_p(i+1) = (nd_{i+1}, \Delta_{i+1})$ where

$$\Delta_{i+1}(x) = \{u_j \mid j \in \{1,\ldots,n\} \wedge t_j = x \wedge u_j \notin \mathcal{V}\} \cup \bigcup_{j \in \{1,\ldots,n\} \wedge t_j = x \wedge u_j \in \mathcal{V}} \Delta_i(u_j)$$

for $x \in Var(nd_i)$.

Example 19 Consider $R = R_5$ and $\mathcal{C} = \{\, g^\sharp(x,x) \to f^\sharp(x,a),\, f^\sharp(c,x) \to g^\sharp(x,b) \,\} \subset \mathrm{DP}(R_5)$. The labeling function for a path $N_1, N_2, N_1, \ldots$ is $L(1) = (N_1, \{x \mapsto \emptyset\})$, $L(2) = (N_2, \{x \mapsto \{a\}\})$, $L(3) = (N_1, \{x \mapsto \{a,b\}\})$, $L(4) = (N_2, \{x \mapsto \{a\}\})$, $L(5) = (N_1, \{x \mapsto \{a,b\}\}), \ldots$, where $N_1 = g^\sharp(x,x) \to f^\sharp(x,a)$ and $N_2 = f^\sharp(c,x) \to g^\sharp(x,b)$. (See Fig. 1)

Definition 20 (Argument Propagation Cycling) Let L_p be a labeling function over $p = nd_1, nd_2, nd_3, \ldots$. We say a finite sequence of labels $L_p(I), L_p(I+1), \ldots, L_p(J)$ is an *argument propagation cycling* (APC for short) if $L_p(I) = L_p(J)$ and the following conditions, called *smoothness condition*, are satisfied for all i $(I \le i < J)$:

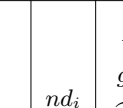
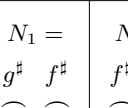
i		1	2	3	4
$L(i)$	nd_i	$N_1 =$	$N_2 =$	$N_3 =$ 	$N_1 =$ 
	$\Delta_i(x)$	$\emptyset$	$\{a\}$	$\{a, b\}$	$\{a\}$

Figure 1: Labeling for Example 19

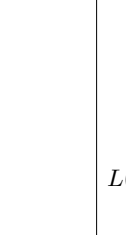
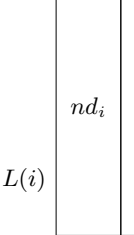
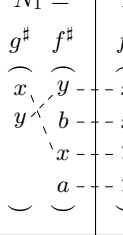
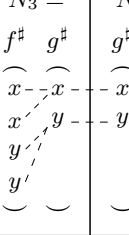
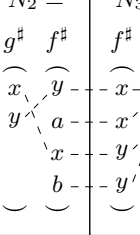
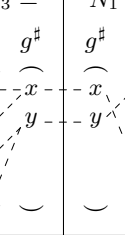
i		1	2	3	4	5	6
$L(i)$	nd_i	$N_1 =$ 	$N_3 =$ 	$N_2 =$ 	$N_3 =$ 	$N_1 =$ 	$N_3 =$ 
	$\Delta_i(x)$	$\emptyset$	$\{b\}$	$\{b\}$	$\{a\}$	$\{a\}$	$\{b\}$
	$\Delta_i(y)$	$\emptyset$	$\{a\}$	$\{a\}$	$\{b\}$	$\{b\}$	$\{a\}$

Figure 2: Labeling for Example 22

1. $\Delta_i(x)$ is joinable for each $x \in Dom(\Delta_i)$.
2. For all j $(1 \leq j \leq n)$ such that $t_j \notin \mathcal{V}$,
   (a) $\Delta_i(u_j)$ is joinable to t_j if $u_j \in \mathcal{V}$;
   (b) u_j is reachable to t_j if $u_j \notin \mathcal{V}$.

where

$$L_p(i) = (v^\sharp \to f^\sharp(u_1, \ldots, u_n), \Delta_i) \text{ and}$$
$$L_p(i+1) = (f^\sharp(t_1, \ldots, t_n) \to s^\sharp, \Delta_{i+1}) \ .$$

We say an APC is *minimal* if all its proper subsequences are not APC.

Example 21 Consider the labeling function L in Example 19. The sequence $L(2)$, $L(3)$, $L(4)$ is a minimal APC.

One may think that every minimal APC contains no repetition of a same node except the edges. However it is not correct in general as shown by the following example.

Example 22 Consider a TRS $R_6 = \{\, g(x,y) \to f(y,b,x,a),\ g(x,y) \to f(y,a,x,b),\ f(x,x,y,y) \to g(x,y) \,\}$ and $\mathcal{C} = \mathrm{DP}(R_6)$. The minimal APC over a path $N_1, N_3, N_2, N_3, N_1, \ldots$ is the sequence $L(2), L(3), \ldots, L(6)$ as shown in Fig. 2 and $L(2), L(3), L(4)$ is not an APC. Indeed $N_3, N_2, N_3, N_2, \ldots$ is not a dependency chain.

Lemma 23 For an eTRS R such that ground reachability and ground joinability are decidable and for a shallow eTRS $\mathcal{C}$, the existence of APC is decidable.

Proof: Firstly we take an approximate graph G of $(R, \mathcal{C})$-graph. The procedure tries searches starting from every node in G. In traversing edges, it quits if an APC is found and backtracks traversal if the path does not satisfy the smoothness condition. The correctness of this procedure is obvious. The range of the labeling function is finite since the possible elements in $\Delta_k(x)$ of the labeling function are ground terms at depth 1 that occurs in the right-hand side of nodes. Since the smoothness condition is decidable by the assumption, termination of the procedure is guaranteed. □

For an APC $(nd_I, \Delta_I), \ldots, (nd_J, \Delta_J)$, $RanD$ denotes union of ranges of all Δ_is contained in the APC, that is $RanD = \mathrm{Ran}(\Delta_I) \cup \mathrm{Ran}(\Delta_{I+1}) \cup \cdots \mathrm{Ran}(\Delta_J)$ where $\mathrm{Ran}(\Delta_i) = \{\Delta_i(x) \mid x \in \mathrm{Dom}(\Delta_i)\}$. From the smoothness condition 1, each set $S \in RanD$ of terms is joinable to a term s, which we denote by a function $\mathcal{E}v : \mathcal{P}(\mathcal{T}(\mathcal{F})) \to \mathcal{T}(\mathcal{F})$ where $\mathcal{E}v(\emptyset)$ is assined by a fresh variable.

We say that a natural extension of $(R, \mathcal{C})$-chain $\cdots nd_{-1}, nd_0, nd_1$ is *backward-infinite*. In order to avoid confusion, we sometimes say that an infinite $(R, \mathcal{C})$-chain is forward-infinite. Next lemma will formally express the relation between an APC and an infinite $(R, \mathcal{C})$-chain.

Lemma 24 Let R be an eTRS and $\mathcal{C}$ be a right-linear shallow eTRS. Then,

1. there exists a forward and backward-infinite $(R, \mathcal{C})$-chain if there exists an APC, and
2. there exists an APC if there exists a forward or backward-infinite $(R, \mathcal{C})$-chain.

Proof: We firstly show the former part. Let $L_p(I), \ldots, L_p(J)$ be an APC over a path $nd_1, \cdots, nd_I, \cdots, nd_J$. The following procedure constructs substitutions $\tau_I, \cdots, \tau_J$ so as to obtain a cyclic $(R, \mathcal{C})$-chain $nd_I\tau_I \to_R^* nd_{I+1}\tau_{I+1} \to_R^* \cdots \to_R^* nd_J\tau_J$.

1. Firstly, do the following repeatedly while applicable, starting with empty substitutions $\tau_i = \emptyset$ $(I \leq i \leq J)$.
   - Let $nd_i = v^\sharp \to f^\sharp(u_1, \ldots, u_n)$ and $nd_{i+1} = f^\sharp(t_1, \ldots, t_n) \to s^\sharp$. Set $\tau_i := \tau_i \cup \{u_j \mapsto t_j\tau_{i+1}\}$ if $I \leq i < J$, $u_j \in \mathcal{V} - \mathrm{Dom}(\tau_i)$ and $t_j\tau_{i+1} \notin \mathcal{V}$ for some $j \in \{1, \ldots, n\}$.
   - Set $\tau_J := \tau_I$ if $\tau_J \neq \tau_I$.
2. Secondly, do the following repeatedly while applicable.
   - Let $nd_i\tau_i = v^\sharp \to f^\sharp(u_1, \ldots, u_n)$ and $nd_{i+1}\tau_{i+1} = f^\sharp(t_1, \ldots, t_n) \to s^\sharp$. Set $\Delta_i := \Delta_i[u_j \mapsto \Delta_{i+1}(t_j)]$ if $I \leq i < J$, $u_j, t_j \in \mathcal{V}$ and $\Delta_i(u_j) \neq \Delta_{i+1}(t_j)$ for some $j \in \{1, \ldots, n\}$, where $\Delta[x \mapsto T]$ denotes a mapping Δ' such that $\Delta'(x) = T$ and $\Delta'(y) = \Delta(y)$ for $y \neq x$.
   - Set $\Delta_J := \Delta_I$ if $\Delta_J \neq \Delta_I$.
3. Thirdly, do the follwing
   - Set $\tau_i := \tau_i \cup \{x \mapsto \mathcal{E}v(\Delta_i(x))\}$ for $x \in Var(nd_i) - Dom(\tau_i)$.

Note that the uniqueness of each substitution τ_i is guaranteed by the right-linearity of nodes. This procedure eventually stops and construct a cyclic $(R, \mathcal{C})$-chain from the smoothness of the APC. Hence the existence of an forward and backward-infinite $(R, \mathcal{C})$-chain is easily shown.

Next, we argue that there exists an APC over a given forward-infinite $(R, \mathcal{C})$-chain. Let the $(R, \mathcal{C})$-chain be $nd_1, nd_2, \ldots$ where $nd_i = t_i^\sharp \to s_i^\sharp \in \mathcal{C}$. There exists an APC (with smoothness condition ignored) over the path. Note that it is also possible even if the given chain is backward-infinite one $\cdots, nd_{-1}, nd_0, nd_1$, since we can choose a natural number N small enough such that an APC can be found along the path $nd_N, nd_{N+1}, \cdots, nd_0, nd_1$. Let the APC be $L(I), \cdots, L(J)$. We have an sequence $t_I^\sharp\tau_I \to_{\mathcal{C}} s_I^\sharp\tau_I \to_R^* t_{I+1}^\sharp\tau_{I+1} \to_{\mathcal{C}} s_{I+1}^\sharp\tau_{I+1} \to_{R\cup\mathcal{C}}^* \cdots \to_{R\cup\mathcal{C}}^* t_J^\sharp\tau_J \to_{\mathcal{C}} s_J^\sharp\tau_J$, where $t_i^\sharp \to s_i^\sharp$ is a rule in $L(i)$. The satisfaction of the smoothness condition follows from the traces of the reductions of ground terms at depth 1 in the sequence. □

The following example is helpful for understanding the first step of the procedure in the above proof.

Example 25 Consider the APC $L(2), L(3), L(4)$ from a path $nd_1, nd_2, \ldots$ (see Fig. 1). We show the existence of cycling reduction sequence $t \to_{R\cup\mathcal{C}}^+ t$. The first step of the procedure in the proof of Lemma 24 produce substitutions $\tau_2 = \tau_3 = \tau_4 = \{x \mapsto c\}$ and the remaining steps do nothing. We have $f^\sharp(c, c) \to_{\mathcal{C}} g^\sharp(c, b) \to_R^* g^\sharp(c, c) \to_{\mathcal{C}} f^\sharp(c, a) \to_R^* f^\sharp(c, c)$ from the sequence $nd_2\tau_2, nd_3\tau_3, nd_4\tau_4$.

The following example is helpful for understanding the second and the third step of the procedure in the above proof.

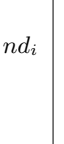
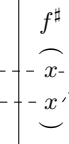
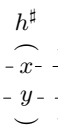
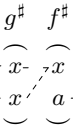
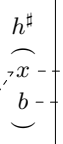
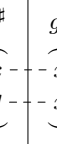
i		1	2	3	4	5	6	7
$L(i)$	nd_i	$N_1 =$	$N_2 =$ 	$N_3 =$ 	$N_1 =$ 	$N_2 =$ 	$N_3 =$ 	$N_1 =$ 
	$\Delta_i(x)$	$\emptyset$	$\{a\}$	$\{a\}$	$\{a,b\}$	$\{a,b\}$	$\{a,b\}$	$\{a,b\}$
	$\Delta_i(y)$			$\{b\}$			$\{b\}$	

Figure 3: Labeling for R_9

Example 26 Consider the following TRS $R_7 = \{\, g(x,x) \to f(x,a),\ f(x,x) \to h(x,b),\ h(x,y) \to g(x,y),\ a \to c,\ b \to c \,\}$ and $\mathcal{C} = \{\, g^\sharp(x,x) \to f^\sharp(x,a),\ f^\sharp(x,x) \to h^\sharp(x,b)\ h^\sharp(x,y) \to g^\sharp(x,y) \,\} \subset \mathrm{DP}(R_7)$. It has an APC $L(4), L(5), L(6), L(7)$ for a labeling function L in Fig. 3. According to the procedure in the proof of Lemma 24, the first step produce $\tau_4 = \tau_5 = \tau_6 = \tau_7 = \emptyset$ and the second step changes Δ_6 so that $\Delta_6(y) = \{a, b\}$. Hence we obtain substitutions $\tau_4 = \tau_5 = \tau_7 = \{x \mapsto c\}$ and $\tau_4 = \{x \mapsto c, y \mapsto c\}$ by the third step since $\mathcal{E}v(\{a,b\}) = c$. Therefore, we have $g^\sharp(c,c) \to_{\mathcal{C}} f^\sharp(c,a) \to_R^* f^\sharp(c,c) \to_{\mathcal{C}} h^\sharp(c,b) \to_R^* h^\sharp(c,c) \to_{\mathcal{C}} g^\sharp(c,c)$ from the sequence $nd_4\tau_4, \ldots, nd_7\tau_7$.

Next, based on Lemma 24, we give proofs for Theorem 5–(2) and (3).

Proof: **(Theorem 5–(2))** Let R be a right-linear shallow TRS. Then, $\mathrm{DP}(R)$ is also right-linear shallow. We know ground reachability and ground joinability of right-linear shallow TRSs are decidable [3, 9, 10, 13]. By Lemma 23, we can decide the existence of APC. Thus we can decide the existence of a forward-infinite $(R, \mathrm{DP}(R))$-chain by Lemma 24. The theorem follows from Theorem 1. □

Proof: **(Theorem 5–(3))** Let R be a left-linear shallow TRS. Then R^{-1} and $\mathrm{DP}(R)^{-1}$ are right-linear shallow eTRSs We know ground reachability and ground joinability of right-linear shallow TRSs are decidable [3, 9, 10, 13]. By Lemma 23, we can decide the existence of APC. If an APC exists, we have a backward-infinite $(R^{-1}, \mathrm{DP}(R)^{-1})$-chain from the former part of Lemma 24, which shows the existence of a forward-infinite $(R, \mathrm{DP}(R))$-chain. If no APC exists, we have no backward-infinite $(R^{-1}, \mathrm{DP}(R)^{-1})$-chain from the latter part of Lemma 24, which shows the non-existence of a forward-infinite $(R, \mathrm{DP}(R))$-chain. The theorem follows from Theorem 1. □

3.3 Combining the two techniques

In this subsection, we combine the techniques in the above two subsections and show the decidability of termination for some larger classes. This is based on the following lemma.

Proposition 27 For TRSs R, $\mathcal{C}$ and $\mathcal{C}'$ such that $\mathcal{C} \supseteq \mathcal{C}'$, the following statements are equivalent.

1. There exists an infinite $(R, \mathcal{C})$-chain.
2. There exists an infinite $(R, \mathcal{C}')$-chain or there exists an infinite $(R, \mathcal{C})$-chain with infinite use of pairs in $\mathcal{C} - \mathcal{C}'$.

Proof: Since the latter implies the former trivially, we show the converse. Suppose we have an infinite $(R, \mathcal{C})$-chain $nd_1, nd_2, \ldots$ with finite use of pairs in $\mathcal{C} - \mathcal{C}'$. Letting nd_n is the last use of a pair in $\mathcal{C} - \mathcal{C}'$, the infinite subsequence $nd_{n+1}, nd_{n+2}, \ldots$ is a $(R, \mathcal{C}')$-chain. □

Definition 28 (Growing TRS) A rewrite rule $l \to r$ is *growing* if all variables in $\mathrm{Var}(l) \cap \mathrm{Var}(r)$ occur at positions with depth 0 or 1 in l. An eTRS R is *growing* if every rewrite rule in R is growing and R is *rev-growing* if R^{-1} is growing.

Example 29 TRS $R_8 = \{\ f(a,x) \to g(x,b),\ g(x,y) \to h(x,p(x,y)),\ h(c,x) \to f(x,x)\,\}$ is left-linear growing.

Proof: **(Theorem 5–(4))** Let R be a right-linear rev-growing TRS with $\mathrm{DP}(R)$ being shallow or right-ground. Let $\mathcal{C}_s$ be the set of all shallow pairs in $\mathrm{DP}(R)$. We know ground reachability and

ground joinability of right-linear rev-growing TRSs are decidable [10, 13]. Since $\mathcal{C}_s$ is right-linear shallow, we can decide the existence of APC by Lemma 23. If an APC exists then we have an infinite $(R, \mathcal{C}_s)$-chain by Lemma 24, which implies that R is non-terminating. Otherwise, from Prop. 27, it is enough to decide the existence of an infinite $(R, \mathrm{DP}(R))$-chain with infinite use of pairs in $\mathrm{DP}(R) - \mathcal{C}_s$, which is a set of right-ground pairs. This is decidable from Lemma 14. □

Proof: **(Theorem 5–(5))** Let R be a left-linear growing TRS with $\mathrm{DP}(R)$ being shallow or right-ground. Let $\mathcal{C}_s$ be the set of all shallow pairs in $\mathrm{DP}(R)$. Then R^{-1} is right-linear rev-growing and $\mathcal{C}_s^{-1}$ is right-linear shallow. Since we know ground reachability and ground joinability of right-linear rev-growing TRSs are decidable [10, 13], we can decide the existence of APC by Lemma 23. If an APC exists then we have a backward-infinite $(R^{-1}, \mathcal{C}_s^{-1})$-chain by Lemma 24, which implies the existence of an infinite $(R, \mathcal{C}_s)$-chain and hence R is non-terminating. Otherwise, from Prop. 27, it is enough to decide the existence of an infinite $(R, \mathrm{DP}(R))$-chain with infinite use of pairs in $\mathrm{DP}(R) - \mathcal{C}_s$, which is a set of right-ground pairs. This is decidable from Lemma 14. □

4 Comparison

In this section, we compare our results with some existing results.

Lemma 30 For a semi-constructor TRS R, the following statements are equivalent:

1. R does not terminate.
2. $\mathrm{DG}(R)$ contains a cycle.

Proof: Suppose R does not terminate. There exists an infinite dependency chain by Theorem 1. Hence the dependency graph must have a cycle, otherwise it causes a contradiction.

Conversely, for every edge from a node $s^\sharp \to t^\sharp$ to a node $u^\sharp \to v^\sharp$ in a cycle, there exists a substitution τ such that $t^\sharp \to_R^* u^\sharp\tau$. Thus we can easily construct an infinite dependency chain. □

Lemma 31 The dependency graph of semi-constructor TRSs is not computable.

Proof: By encoding Post's Correspondence Problem. Let $\{\langle u_i, v_i\rangle \in \Sigma^+ \times \Sigma^+ \mid 1 \le i \le n\}$ be a finite set of PCP pairs.

$$
\begin{aligned}
TRS\, R_9 = \; & \{\varepsilon \to e_i(\varepsilon) \mid 1 \le i \le n\} \cup \\
& \{\varepsilon \to f(c,d)\} \cup \\
& \{b \to a(b),\ b \to a(\varepsilon) \mid b \in \{c,d\},\ a \in \Sigma\} \cup \\
& \{f(x,x) \to g(x,x)\} \cup \\
& \{e_i(g(u_i(x), v_i(y))) \to g(x,y) \mid 1 \le i \le n\} \cup \\
& \{e_i(g(u_i(\varepsilon), v_i(\varepsilon))) \to \varepsilon \mid 1 \le i \le n\}
\end{aligned}
$$

Defined symbol of R_9 is $\{\varepsilon, c, d, f\} \cup \{e_i \mid 1 \le i \le n\}$, R_9 is a semi-constructor TRS and it is a variant of the example in [12]. Notice that the following statement is true: in $\mathrm{DG}(R_9)$, there is an edge from node $\varepsilon^\sharp \to e_1^\sharp(\varepsilon)$ to node $e_1^\sharp(g(u_1(\varepsilon), v_1(\varepsilon))) \to \varepsilon^\sharp$ if and only if PCP has a solution. □

Note that reachability problem is undecidable for linear semi-constructor TRSs [11]. However this fact is not enough to prove the above lemma because the use of reachability in dependency graphs are limited.

In the reference [9], Middeldorp proposed a decision procedure for termination of right-ground TRSs which is dependency graph based. Denoting growing approximation dependency graph by $\mathrm{DG}_\mathrm{g}(R)$, he showed that for right-ground TRS R, $\mathrm{DG}(R) = \mathrm{DG}_\mathrm{g}(R)$, that is, the dependency graph of the right-ground TRS is computable. Thus, the decision procedure proposed is that: compute the dependency graph of R using the growing approximation and then check the existence of cycles. For semi-constructor case, we also have Lemma 30 to assure that semi-constructor TRS terminates if and only if there is no cycles in the dependency graph. However, the dependency graph based method can not be applied to semi-constructor case, since its dependency graph is not computable by Lemma 31.

The following theorem shown by Godoy and Tiwari [5] is also given as a corollary of Theorem 5–(4) since TRSs in this class satisfy the assumption of our theorem.

Theorem 32 ([5]) Termination of TRSs that consist of right-linear shallow rules, collapsing rules and right-ground rules is decidable.

Nagaya and Toyama [10] obtained the decidability result for almost orthogonal growing TRSs. We claim that the applicable classes of Nagaya's method and ours do not cover each other. Considering R_4 in Example 17 and R_8 in Example 29: R_4 is left-linear shallow, but it is not almost orthogonal since there is a non-trivial critical pair $\langle f(g(a), z), f(z, b)\rangle$; $\mathrm{DP}(R_8)$ does not fit either of the applicable classes we proposed, but it is orthogonal.

5 Conclusion

One research direction on termination is to find more general classes of TRSs whose termination is decidable. We proposed several positive results listed in Theorem 5.

We propose some conjectures and list them as follows:

Conjecture 33 Termination of right-linear rev-growing TRSs with all the dependency pairs being left-linear is decidable.

Conjecture 34 Termination of left-linear growing TRSs with all the dependency pairs being right-linear is decidable.

Conjecture 35 Termination of shallow TRSs is undecidable.

Acknowledgement

We would like to thank Ashish Tiwari and the anonymous referees for their helpful comments and remarks. We are particularly grateful to one of anonymous referees who indicated the idea for combining our methods (Prop. 27). This work is partly supported by MEXT.KAKENHI #18500011 and #16650005.

References

[1] T. Arts and J. Giesl: Termination of term rewriting using dependency pairs. Theoretical Computer Science **236** (2000) 133–178.

[2] F. Baader and T. Nipkow: Term rewriting and all that. Cambridge University Press (1998).

[3] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison and M. Tommasi: Tree automata techniques and applications (1997). `http://www.grappa.univ-lille3.fr/tata/`

[4] N. Dershowitz: Termination of linear rewriting systems. Proc. of the 8th international colloquium on automata, languages and programming, LNCS **115** Springer-Verlag (1981) 448–458.

[5] G. Godoy and A. Tiwari: Termination of rewrite systems with shallow right-linear, collapsing and right-ground rules. Proc. of the 20th international conference on automated deduction, LNAI **3632** (2005) 164–176.

[6] N. Hirokawa and A. Middeldorp: Dependency pairs revisited. Proc. of the 15th international conference on rewriting techniques and applications, LNCS **3091** (2004) 249–268.

[7] G. Huet and D. Lankford: On the uniform halting problem for term rewriting systems. Technical Report **283** INRIA (1978).

[8] F. Jacquemard: Decidable approximations of term rewriting systems. Proc. of the 7th international conference on rewriting techniques and applications, LNCS **1103** (1996) 362–376.

[9] A. Middeldorp: Approximating dependency graphs using tree automata techniques. Proc. of the international joint conference on automated reasoning, LNAI **2083** (2001) 593–610.

[10] T. Nagaya and Y. Toyama: Decidability for left-linear growing term rewriting systems. Proc. of the 10th international conference on rewriting techniques and applications, LNCS **1631** (1999) 256–270.

[11] I. Mitsuhashi, M. Oyamaguchi, Y. Ohta and T. Yamada. The joinability and unification problems for confluent semi-constructor TRSs. Proc. of the 15th international conference on rewriting techniques and applications, LNCS **3091** (2004) 285–300.

[12] I. Mitsuhashi, M. Oyamaguchi and T. Yamada: The reachability and related decision problems for monadic and semi-constructor TRSs. Information Processing Letters **98** (2006) 219–224.

[13] T. Takai, Y. Kaji and H. Seki: Right-linear finite path overlapping term rewriting systems effectively preserve recognizability. Proc. of the 11th international conference on rewriting techniques and applications, LNCS **1833** (2000) 246–260.