Decidability of Innermost Termination for Semi-Constructor Term Rewriting Systems

Keita Uchiyama †, Masahiko Sakai ‡, Naoki Nishida †, Toshiki Sakabe † and Keiichirou Kusakari †

†School of Engineering, Nagoya University
‡Graduate School of Information Science, Nagoya University

{uchiyama@sakabe.i.is.nagoya-u.ac.jp, sakai,nishida,sakabe,kusakari}@is.nagoya-u.ac.jp

Abstract

Yi and Sakai [7] showed that the termination problem is decidable for a class of semi-constructor term rewriting systems, which is a superclass of the class of right ground term rewriting systems. The decidability was shown by the fact that every non-terminating TRS in the class has a loop. In this paper we modify the proof of [7] to show that innermost termination is decidable for the class of semi-constructor TRSs.

1 Introduction

Termination is one of the central properties of term rewriting systems (TRSs for short), where we say a TRS terminates if it does not admit any infinite reduction sequence. Since termination is undecidable in general, several decidable classes have been studied [3, 4, 5, 6, 7]. The class of semi-constructor TRSs showed in [7] is one of them, where a TRS is in the class if every subterm of right hand sides of rules, whose root symbol is defined, is ground.

Innermost reduction, the strategy which rewrites innermost redexes, is used as call-by-value computation semantics. The termination property with respect to innermost reduction is called innermost termination. Since innermost termination is also undecidable in general, methods for proving innermost termination have been studied [1].

In this paper, we prove that innermost termination for semi-constructor TRSs is a decidable property. The proof is done by following the proof [7] for decidability of termination for semi-constructor TRSs.

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: $\text{arity}: \mathcal{F} \to \mathbb{N} = \{0, 1, 2, \ldots\}$. 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 $T(\mathcal{F}, \mathcal{V})$. The set of ground terms is denoted by $T(\mathcal{F}, \emptyset)$ ($T(\mathcal{F})$ for short). The set of variables occurring in a term $t$ is denoted by $\text{Var}(t)$. We write $s = t$ when two terms $s$ and $t$ are identical. The root symbol of a term $t$ is denoted by $\text{root}(t)$.

Let $C$ be a context with a hole $\square$. We write
C[t] for the term obtained from C by replacing □ with a term t. We say t is a subterm of s if s = C[t] for some context C. We denote the subterm ordering by ≤, that is, t ≤ s if t is a subterm of s, and t < s if t ≤ s and t ≠ s.

A substitution θ is a mapping from V to T(\mathcal{F}, \mathcal{V}) such that the set \text{Dom}(θ) = \{x ∈ V | θ(x) ≠ x\} is finite. We usually identify a substitution θ with the set \{x → θ(x) | x ∈ \text{Dom}(θ)\} of variable bindings. In the following, we write θt instead of θ(t).

A rewrite rule l → r is a directed equation which satisfies l ∉ V and \text{Var}(r) ⊆ \text{Var}(l). A term rewriting system TRS is a finite set of rewrite rules. A redex is a term lθ for a rule l → r and a substitution θ. A term containing no redex is called a normal form. A redex is innermost, if its all proper subterms are in normal forms. A substitution θ is normal if θx is in normal forms for every x. The reduction relation \rightarrow_R ⊆ T(\mathcal{F}, \mathcal{V}) × T(\mathcal{F}, \mathcal{V}) associated with a TRS R is defined as follows: s →_R t if there exist a rewrite rule l → r ∈ R, a substitution θ, and a context C such that s = C[lθ] and t = C[rθ]. We say that s is reduced to t by contracting redex lθ. Especially, if s is reduced to t by contracting an innermost redex, then s →_R t is said to be innermost reduction denoted by s →^*_R t.

Let → be a binary relation on terms, the transitive closure of → is denoted by →+. The transitive and reflexive closure of → is denoted by →^*. If s →^* t, then we say that there is a reduction sequence starting from s to t or t is reachable from s with respect to →. We write s →^k t if t is reachable from s by reductions with k steps. For a TRS R, a term t ∈ T(\mathcal{F}, \mathcal{V}) terminates with respect to → if there is no infinite reduction sequence starting from t with respect to →. We say that R terminates with respect to → if every term terminates with respect to →. We say that R innermost terminates if it terminates with respect to →^*_R.

**Proposition 1** For a TRS R, if there is a reduction s →^*_R t, then C[s] →^*_R C[t] for any context C.

For a TRS R, a function symbol f ∈ \mathcal{F} is a defined symbol of R if f = root(l) for some rule l → r ∈ R. The set of all defined symbols of R is denoted by \text{D}_R = \{\text{root}(l) | l → r ∈ R\}. A term t has a defined root symbol if root(t) ∈ \text{D}_R.

Let R be a TRS over a signature \mathcal{F}. \mathcal{F}^\ddagger denotes the union of \mathcal{F} and \text{D}_R^\ddagger = \{f^\ddagger | f ∈ \text{D}_R\} where \mathcal{F} ∩ \text{D}_R^\ddagger = \emptyset and \mathcal{F}^\ddagger has the same arity as f. We call these fresh symbols dependency pair symbols. Given a term t = f(t_1, ..., t_n) ∈ T(\mathcal{F}, \mathcal{V}) with f defined, we write t^\ddagger for the term f^\ddagger(t_1, ..., t_n). If l → r ∈ R and u is a subterm of r with a defined root symbol, then the rewrite rule l^\ddagger → u^\ddagger is called a dependency pair of R. The set of all dependency pairs of R is denoted by \text{DP}(R).

### 3 Decidability of Innermost Termination for Semi-Constructor TRSs

Many infinite reduction sequences contain loops, in which an instance of a term re-occurs as a subterm. Decidability of termination for semi-constructor TRSs is proved based on the observation that there exists an infinite reduction sequence having a loop if it is not terminating[7]. In this section, we prove decidability of innermost termination in similar way.

**Definition 2 (Loop)** A reduction sequence loops if it contains t^' →^+ C[t^'θ] for some context C, substitution θ. Similarly, a reduction sequence cycles if containing t^' →^+ C[t'], and head-cycles if containing t^' →^+ t'.

The following proposition holds from proposition 1.

**Proposition 3** If there exists a cycling sequence that loops, then there exists infinite innermost sequence.

**Definition 4 (Semi-Constructor TRS)** A term t ∈ T(\mathcal{F}, \mathcal{V}) is a semi-constructor term if every term s such that s ⊆ t and root(s) ∈ \text{D}_R is ground. A TRS R is a semi-constructor system if r is a semi-constructor term for every rule l → r ∈ R.
A TRS $R$ is called right-ground if for every $l \rightarrow r \in R$, $r$ is ground.

**Proposition 5** The following statements hold:

1. Right-ground TRSs are semi-constructor systems.

2. For a TRS $R$, $R$ is semi-constructor TRS if and only if all rules in $\text{DP}(R)$ are right-ground.

For a given TRS, let $T_{\infty}$ denote the set of all minimal non-terminating terms for $\rightarrow$, here "minimal" is used in the sense that all its proper subterms terminate. For a given TRS, let $N^F$ denote the set of all normal forms for $\rightarrow$.

**Definition 6 (Innermost DP-chain)** For a TRS $R$, a sequence of the elements of $\text{DP}(R)$

\[ s_1^R \rightarrow s_1^*, s_2^R \rightarrow s_2^*, \ldots \text{ is an innermost dependency chain if there exist substitutions} \]

\[ \tau_1, \tau_2, \ldots \text{ such that } s_i^R \tau_i \in N^F \text{ in } R \text{ and } t_i^R \rightarrow s_i^R \tau_{i+1} \tau_{i+1} \text{ holds for every } i. \]

**Theorem 7 ([1])** For a TRS $R$, $R$ does not innermost terminate if and only if there exists an infinite innermost dependency chain.

**Definition 8 (C-min)** For a TRS $R$, let $C \subseteq \text{DP}(R)$. An infinite reduction sequence in $R \cup C$ in the form $t_i^R \rightarrow s_i^R \tau_i \in R \cup C$ with $t_i \in T_{\infty}$ for all $i \geq 1$ is called a $C$-min innermost reduction sequence. We use $C_{\min}(t^R)$ to denote the set of all $C$-min innermost reduction sequence starting from $t^R$.

**Proposition 9 ([1])** Given a TRS $R$, we have the following statements:

1. If there exists an infinite innermost dependency chain, then $C_{\min}(t^R) \neq \emptyset$ for some $C \subseteq \text{DP}(R)$ and $t \in T_{\infty}^R$.

2. For any sequence in $C_{\min}(t^R)$, reduction by rules of $R$ takes place below the root while reduction by rules of $C$ takes place at the root.

3. For any sequence in $C_{\min}(t^R)$, there is at least one rule in $C$ which is applied infinitely often.

**Lemma 10** For any sequence in $C_{\min}(t^R)$, subsequence $s_i^R \rightarrow s_{i+1}^R \rightarrow s_{i+2}^R \rightarrow \ldots$ implies $s \rightarrow_R C[s^R]$ for some context $C$.

**Proof** We use induction on number $n$ of reduction steps $s_i^R \rightarrow s_{i+1}^R \rightarrow \ldots$. In the case that $n = 0$, it holds with $C = \emptyset$. Suppose $n \geq 1$. Then we have $s_i^R \rightarrow s_{i+1}^R \rightarrow s_{i+2}^R \rightarrow \ldots$ for some $s^R$. By induction hypothesis, $s \rightarrow_R C[s^R]$.

(i): Consider the case that $s''^R \rightarrow s^R$. Since $s''^R \rightarrow_R s^R$, we have $C[s''] \rightarrow_R C[s^R]$ by Proposition 1. Hence $s \rightarrow_R C[s^R]$.

(ii): Consider the case that $s''^R \rightarrow_R C[s'']$. By definition of dependency pair, $C[s''] \rightarrow_R C[C'[s'']]$, by Proposition 1. Hence $s \rightarrow_R C[C'[s'']]$.

**Lemma 11** For a TRS $R$, if $sq \in C_{\min}(t^R)$, then $sq$ head-cycles.

**Proof** Let $sq \in C_{\min}(t^R)$, then there is a subsequence $t_i^R \rightarrow \ldots \rightarrow C[s'^R]$ in $sq$. From Proposition 9-(2) and the fact that dependency pair symbols appears only in dependency pairs, we have $C[t_k^R] = t_k^R$, which implies that $sq$ head-cycles.

**Lemma 12** For a TRS $R$, if $sq \in C_{\min}(t^R)$, then there is a term $t_k^R$ in $sq$ such that sequence starting from $t_k^R$ cycles with respect to $\rightarrow_R$.

**Proof** From Lemma 11 and Lemma 10.

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

1. $R$ does not innermost terminate.

2. There exists $l^R \rightarrow u^R \in \text{DP}(R)$ such that $sq$ head-cycles for some $sq \in C_{\min}(t^R)$.

**Proof** (2 $\Rightarrow$ 1): It is obvious from Lemma 12, and Proposition 3. (1 $\Rightarrow$ 2): By Theorem 7 there exists an infinite innermost dependency chain. By Proposition 9-(1), there exists a sequence $sq \in C_{\min}(t^R)$. By Proposition 9-(2,3),
there is some rule $v^i \rightarrow u^i \in C$ which is applied at root reduction in $sq$ infinitely often. By Proposition 5–2, $u^i$ is ground. Thus $u^i$ cycle in the form $u^i \xrightarrow{\ast} \Delta \rightarrow_{R,DP(R)} \rightarrow \{f \rightarrow u\} u^i$.

**Theorem 14** Innermost termination of semi-constructor TRSs is decidable.

**Proof** The decision procedure for termination of semi-constructor TRS $R$ with innermost rewriting is as follows: consider all terms $u_1, u_2, \ldots, u_n$ corresponding to the right-hand sides of $DP(R) = \{v^i \rightarrow u^i | 1 \leq i \leq n\}$, and simultaneously generate all innermost reduction sequences with respect to $R$ starting from $u_1, u_2, \ldots, u_n$. It halts if it enumerates all reachable terms exhaustively or it detects a cycling reduction sequence $u_i \rightarrow_R C[u_i]$ for some $i$.

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

4 Some Extension and Example

In this section, we relax the condition that guarantees decidability of innermost termination for semi-constructor TRSs.

**Lemma 15** Let $R$ be a TRS whose innermost termination is equivalent to the non-existence of an innermost dependency chain that contains infinite use of right-ground dependency pairs. Then innermost termination of $R$ is decidable.

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

**Definition 16** (Innermost DP-Graph [1]) The innermost dependency graph of a TRS $R$ is directed graph whose nodes are the dependency pairs and there is an arc from $s^i \rightarrow t^i$ to $u^i \rightarrow u^i$ if there exists a normal substitution $\sigma, \tau$ such that $t^i \sigma \rightarrow \ast u^i \tau$ and $u^i \tau$ is in normal forms with respect to $R$.

An approximated dependency graph is a graph that contains innermost dependency graph as subgraph. One of computable such graphs are proposed [1].

**Theorem 17** Let $R$ be a TRS and $G$ be an approximated dependency graph of $R$. If at least one node in the cycle is right-ground for every cycle of $G$, then innermost termination of $R$ is decidable.

**Proof** From Lemma 15.

**Example 18** Let $R = \{f(s(x)) \rightarrow g(x), g(s(x)) \rightarrow f(s(0))\}$. Then $DP(R) = \{f^\#(s(x)) \rightarrow g^\#(x), g^\#(s(x)) \rightarrow f^\#(s(0))\}$. The dependency graph of $R$ has one cycle, which contains a right ground node. The termination of $R$ is decidable by Theorem 17. Actually we know $R$ is innermost terminating from the procedure in the proof of Theorem 14 since all innermost reduction sequences from $f(s(0))$ terminates.

References


[3] N. Dershowitz. Termination of Linear Rewriting Systems. In Eighth International Colloquium on Automata, Languages and


