Root Stabilisation Using Dependency Pairs

Jörg Endrullis[1] and Jeroen Ketema[2]

[1] Department of Computer Science, Vrije Universiteit Amsterdam
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
joerg@few.vu.nl
[2] Research Institute of Electrical Communication, Tohoku University
2-1-1 Katahira, Aoba-ku, Sendai 980-8577, Japan
jketema@nue.riec.tohoku.ac.jp

1 Introduction

A dependency pair problem [1] is elegantly formulated as a relative termination problem [2]: For a TRS $\mathcal{R} = (\Sigma, R)$, does $\to_{\mathcal{DP}(R)}$ terminate relative to $\to_R$ when the steps from $\to_{\mathcal{DP}(R)}$ occur as *root steps* and the steps from $\to_R$ occur as *non-root steps*? In other words, is $\to_{\mathcal{DP}(\mathcal{R})}$ *root stabilising relative to* $\to_{\mathcal{R}}$?

The current paper is concerned with the application of a number of methods from the dependency pair approach — reduction pairs, dependency graphs, and labelling of defined symbols — in the more general setting of relative root stabilisation. This is interesting for at least three reasons:

- Enhancement of known techniques for proving relative root stabilisation, e.g. matrix interpretations [2].
- Identification of those methods from the dependency pair approach which may have wider application.
- Strong normalisation [3] of finite terms in infinitary TRSs with finite sets of rules, which is root stabilisation in case of left-linear systems due to compression, i.e. every reduction is equivalent to one of length at most ω [4].

2 Preliminaries

We assume familiarity with rewriting [4] and dependency pairs [1]. Below, Σ denotes a signature, V denotes a countable set of variables, and $\mathcal{T}er(\Sigma, V)$ denotes the set of terms over Σ and V; $\mathcal{R} = (\Sigma, R)$ and $\mathcal{S} = (\Sigma, S)$ denote arbitrary TRSs. A *reduction pair* is a pair $(\gtrsim, \succ)$ with $\gtrsim$ a quasi-rewrite order and $\succ$ a stable, not necessarily monotonic, well-founded order compatible with $\gtrsim$.

A binary relation $\to_1$ is called *terminating relative to* $\to_2$ iff $\mathsf{SN}(\to_1/\to_2)$, i.e. $\mathsf{SN}(\to_2^* \cdot \to_1 \cdot \to_2^*)$. A step $\to$ is a *root step* $\xrightarrow{\wedge}$ if it contracts a redex at the root, otherwise it is a *non-root step* $\xrightarrow{\vee}$. Obviously, $\to = \xrightarrow{\wedge} \uplus \xrightarrow{\vee}$. A TRS $\mathcal{R}$ is *root stabilising relative to* $\mathcal{S}$ iff $\mathsf{SN}(\xrightarrow{\wedge}_R/\xrightarrow{\vee}_S)$; $\mathcal{R}$ is *root stabilising* iff it is root stabilising relative to itself, i.e. no reductions with infinitely many root steps occur. A term is *root stable* if no root steps occur in any reduction starting from the term.

Remark 2.1. In a root stabilising TRS, a term may admit infinite reductions and it may require an arbitrary, finite number of root steps to obtain a root stable term. For example, the TRS with the rules $\{a \to g(a),\ f(h(x)) \to f(x),\ g(a) \to h(b),\ g(h(x)) \to h(h(x))\}$ is root stabilising and admits an infinite reduction starting from $f(a)$:

$$f(a) \to f(g(a)) \to f(g^2(a)) \to \cdots \to f(g^n(a)) \to \cdots .$$

For each $n \in \mathbb{N}$, $f(g^n(a))$ is reducible to $f(h^n(b))$ and n root steps are required to reduce $f(h^n(b))$ to the root stable term $f(b)$.

3 Reduction Pairs

We relate relative root stabilisation and reduction pairs. We have the following theorem, which is close to the main theorem of the dependency pair approach [1]:

Theorem 3.1. *It holds that* $\mathsf{SN}(\hat{\to}_R/\check{\to}_S)$ *iff there exists a reduction pair* $(\succsim, \succ)$ *such that* $S \subseteq \succsim$ *and* $R \subseteq \succ$.

Note that $\mathcal{R}$ is root stabilising iff there exists a reduction pair $(\succsim, \succ)$ such that $R \subseteq \succsim \cap \succ$. As usual, we like to weaken proof obligations step-by-step:

Theorem 3.2. *Let* $(\succsim, \succ)$ *be a reduction pair with* $S \subseteq \succsim$ *and* $R \subseteq \succsim \cup \succ$. *It holds that* $\mathsf{SN}(\hat{\to}_R/\check{\to}_{\mathcal{S}})$ *iff* $\mathsf{SN}(\hat{\to}_{R-\succ}/\check{\to}_S)$.

Example 3.3. Consider $R = \{f(g(x), y) \to f(x, f(g(x), y))\}$ [5, Example 4.4], we show root stabilisation, i.e. $\mathsf{SN}(\hat{\to}_R/\check{\to}_R)$. Using the polynomial interpretation $|f(x_1, x_2)| = |x_1|$ and $|g(x)| = 1 + |x|$ over the natural numbers, we obtain $|f(g(x), y)| = 1 + |x| > |x| = |f(x, f(g(x), y))|$. Thereby, $R \subseteq \succsim \cap \succ$ and, hence, R is root stabilising.

Given that argument filterings are defined as usual [1, 6] and that π is an arbitrary argument filtering, write the following:

$$\pi(R) = \{\pi(\ell) \to \pi(r) \mid \ell \to r \in R,\ \pi(\ell) \neq \pi(r)\} .$$

Theorem 3.4. *Let* π *be an argument filtering and* $(\succsim, \succ)$ *a reduction pair with* $R' = \{\ell \to r \in R \mid \pi(\ell) \succ \pi(r)\}$ *and* $\pi(S) \cup \pi(R - R') \subseteq \succsim$. *It holds that* $\mathsf{SN}(\hat{\to}_R/\check{\to}_S)$ *iff* $\mathsf{SN}(\hat{\to}_{R-R'}/\check{\to}_S)$.

The proofs of all the above theorems are standard [1, 2].

4 Dependency Graphs

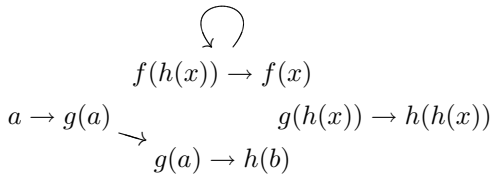
The 'dependency pairs' of a root stabilisation problem $\mathsf{SN}(\hat{\to}_R/\check{\to}_S)$ are the rules of $\mathcal{R}$. The dependency graph provides information on the order in which $\hat{\to}_R$ steps can occur within a $\hat{\to}_R \cup \check{\to}_S$-reduction.

Definition 4.1. *The* dependency graph $\mathcal{DG}(\hat{\to}_R/\check{\to}_S)$ *is a graph that has as its nodes the rewrite rules of* $\mathcal{R}$ *and there is an edge from* $s \to_R t$ *to* $u \to_R v$ *iff there are substitutions* σ *and* τ *with* $\sigma(t) \check{\to}_S^* \tau(u)$*. A cycle* $\mathcal{C}$ *is a nonempty set of nodes such that there is a nonempty path from* n *to* m *for all* $n, m \in \mathcal{C}$*.*

An estimated dependency graph *of* $\mathcal{R}$ *and* $\mathcal{S}$ *is a graph that has as its nodes the rewrite rules of* $\mathcal{R}$ *and that has* $\mathcal{DG}(\hat{\to}_R/\check{\to}_S)$ *as a subgraph.*

The dependency graph is uncomputable in general. In case of dependency pair problems $\mathsf{SN}(\hat{\to}_{\mathcal{DP}(\mathcal{R})}/\check{\to}_R)$, the above definition coincides with the one from the dependency pair approach.

Example 4.2. Denote the set of rules from Remark 2.1 by R. The dependency graph $\mathcal{DG}(\hat{\to}_R/\check{\to}_R)$ is:



Definition 4.3. *Let* $\mathcal{C} \subseteq R$*. An infinite* $\hat{\to}_{\mathcal{C}} \cup \check{\to}_S$*-reduction is called* $\mathcal{C}$-minimal *if all rules from* $\mathcal{C}$ *are applied infinitely often.*

Clearly the existence of a $\mathcal{C}$-minimal rewrite sequence implies that $\mathcal{C}$ is a cycle in the (estimated) dependency graph. Therefore, we have the following:

Theorem 4.4. *It holds that* $\mathsf{SN}(\hat{\to}_R/\check{\to}_S)$ *iff there is no cycle* $\mathcal{C}$ *in the (estimated) dependency graph for which there exists an* $\mathcal{C}$*-minimal rewrite sequence.*

In practice, considering *strongly connected components* [7] is preferred, as the number of cycles can grow exponentially with the number of nodes. A strongly connected component is a maximal cycle (with respect to inclusion).

Theorem 4.5. *It holds that* $\mathsf{SN}(\hat{\to}_{\mathcal{R}}/\check{\to}_{\mathcal{S}})$ *iff for each strongly connected component* $\mathcal{C}$ *in the (estimated) dependency graph* $\mathsf{SN}(\hat{\to}_{\mathcal{C}}/\check{\to}_{\mathcal{S}})$*.*

We use Theorem 4.5 together with dependency graph approximations [1, 7] to split proof obligations into simpler subtasks. These subtasks are root stabilisation problems to which we can apply Theorem 3.2 or 3.4 and Theorem 4.5, recursively.

Example 4.6. We revisit Example 4.2 and show $\mathsf{SN}(\hat{\to}_R/\check{\to}_R)$. The dependency graph contains only one strongly connected component $\mathcal{C} = \{f(h(x)) \to f(x)\}$. Using Theorem 4.5 the proof obligation reduces to $\mathsf{SN}(\hat{\to}_{\mathcal{C}}/\check{\to}_R)$. We choose an interpretation over the ordinal numbers[3] from $\omega \cdot 2$: $|a| = \omega$, $|b| = 0$, $|f(x)| = |x|$, $|g| = I_g(|x|)$ and $|h(x)| = |x| + 1$, where $I_g(\alpha) = \alpha + 1$ for $\alpha \neq \omega$ and $I_g(\omega) = \omega$. Thereby, $\mathcal{C} \subseteq \succ$ and $\mathcal{R} \subseteq \succsim$ and we have $\mathsf{SN}(\hat{\to}_{\mathcal{C}}/\check{\to}_R)$ by Theorem 3.2.

[3] An interpretation over $\mathbb{N}$ is not feasible, because $f(a)$ allows for an arbitrary number of root steps (see Remark 2.1).

5 Labelling Root Symbols

We define a *labelling of root symbols* for $\mathsf{SN}(\hat{\to}_R/\check{\to}_S)$. The *defined symbols* of $\mathcal{R}$ are those from $\mathcal{D} = \{root(\ell) \mid \ell \to r \in R\}$. For every $f \in \mathcal{D}$, let $f^\sharp$ be a fresh function symbol of same arity as f. If $t = f(t_1, \ldots, t_n)$, then $t^\sharp$ stands for $f^\sharp(t_1, \ldots, t_n)$.

Definition 5.1. *The* root labelling $\hat{\mathcal{L}}(R)$ *is defined as follows:*

$$\hat{\mathcal{L}}(R) = \bigcup_{\ell \to r \in R} \begin{cases} \{\ell^\sharp \to r^\sharp\} & \text{if } root(r) \in \mathcal{D} \\ \{\tau(\ell)^\sharp \to \tau(r)^\sharp \mid f \in \mathcal{D},\, \tau = \sigma_{\ell \to x,f}\} & \text{if } r = x \in V \end{cases}$$

where $\sigma_{\ell \to x,f} = \{x \mapsto f(x_1, \ldots, x_n)\}$ *with* $x_1, \ldots, x_n$ *pairwise different variables that do not occur in* ℓ*.*

Observe the close analogy with $\mathcal{DP}(R)$ from the dependency pairs approach. The definition has a case distinction, as rules might be collapsing. A similar case distinction occurs in [8], which deals with context sensitive dependency pairs.

Example 5.2. Consider $R = \{f(x) \to g(f(x)),\ g(f(x)) \to x\}$, we have:

$$\hat{\mathcal{L}}(R) = \{f^\sharp(x) \to g^\sharp(f(x)),\ g^\sharp(f(f(y))) \to f^\sharp(y),\ g^\sharp(f(g(y))) \to g^\sharp(y)\}\ .$$

We have the following theorem:

Theorem 5.3. *It holds that* $\mathsf{SN}(\hat{\to}_R/\check{\to}_S)$ *iff* $\mathsf{SN}(\hat{\to}_{\hat{\mathcal{L}}(R)}/\check{\to}_S)$.

The labelling separates the defined symbols of $\mathcal{R}$ from the symbols of $\mathcal{S}$. Hence, different interpretations may be assigned to the symbols. Moreover, all rules with a non-defined symbol at the root of the right-hand side are thrown away. This increases chances that root-stabilisation may be proved.

6 Subterm and Usable Rules Criteria

The subterm criterion of [9] and the usable rules criterion of [1] are no longer valid in the case of root stabilisation.

The subterm criterion states that for any cycle $\mathcal{C}$ in the dependency graph, if there exists a simple projection $\pi : \mathcal{D}_\mathcal{C} \to \mathbb{N}$ for the defined symbols in $\mathcal{C}$ such that $\pi(u) \trianglerighteq \pi(v)$ for all $u \to v \in \mathcal{C}$ and $\pi(u) \triangleright \pi(v)$ for at least one $u \to v \in \mathcal{C}$, then there is no $\mathcal{C}$-minimal rewrite sequence.

Consider the root stabilisation problem $\mathsf{SN}(\hat{\to}_{\hat{\mathcal{L}}(R)}/\check{\to}_R)$ with:

$$R = \{a \to g(a),\ f(g(x)) \to f(x)\} \qquad \hat{\mathcal{L}}(R) = \{f^\sharp(g(x)) \to f^\sharp(x)\}$$

The dependency graph obviously has a cycle. Given the simple projection $\pi(f^\sharp) = 1$, we obtain $\pi(f^\sharp(g(x))) = g(x) \triangleright x = \pi(f^\sharp(x))$. Hence, the subterm criterion is satisfied. However, we have a $\mathcal{C}$-minimal rewrite sequence:

$$f^\sharp(g(a)) \to f^\sharp(a) \to f^\sharp(g(a)) \to f^\sharp(a) \to f^\sharp(g(a)) \to \cdots\ .$$

The above example also shows that the usable rules criterion is not valid. The rule $a \to g(a)$ would not be considered as usable, but the remaining TRS is root stabilising.

7 Experimental Results

We implemented the above methods as part of Jambox. To analyse their effectiveness, we considered part of the Termination Problem Database 2006 (TPDB): the 103 non-terminating and 76 unknown problems from the TRS category of the Termination Competition 2006 [10], i.e. 179 TRSs in total.

Theorem 3.2 in combination with matrix interpretations [2] allowed to show root stabilisation of 42 of the TRSs.[4] Thereby, 5 proofs depended on matrix interpretations of dimensions 2 or 3; for the remaining 37 problems linear polynomial interpretations sufficed. Preprocessing the problems by labelling the root symbols (Theorem 5.3) increased the score to 46. Finally, using dependency graph approximations [7,11] and Theorem 4.5 often reduced the hardness of the problem, allowing for smaller matrix dimensions and allowing for 47 TRSs to be shown root stabilising. Of the 47 TRSs, 38 are known to be non-terminating and 9 remained unknown in the Termination Competition 2006. The generated proofs are available via: `http://infinity.few.vu.nl/wst07/`.

References

1. Arts, T., Giesl, J.: Termination of term rewriting using dependency pairs. TCS **236** (2000) 133–178
2. Endrullis, J., Waldmann, J., Zantema, H.: Matrix interpretations for proving termination of term rewriting. In: IJCAR'06. Volume 4130 of LNAI. (2006) 574–588
3. Klop, J.W., de Vrijer, R.: Infinitary normalization. In Artëmov, S.N., Barringer, H., d'Avila Garcez, A.S., Lamb, L.C., Woods, J., eds.: We Will Show Them: Essays in Honour of Dov Gabbay. Volume 2. College Publications (2005) 169–192
4. Terese, ed.: Term Rewriting Systems. Cambridge University Press (2003)
5. Lucas, S.: Termination of context-sensitive rewriting by rewriting. In: ICALP'96. Volume 1099 of LNCS. (1996) 122–133
6. Kusakari, K., Nakamura, M., Toyama, Y.: Argument filtering transformation. In: PPDP'99. Volume 1702 of LNCS. (2004) 47–61
7. Hirokawa, N., Middeldorp, A.: Automating the dependency pair method. I&C **199** (2005) 172–199
8. Alarcón, B., Gutiérrez, R., Lucas, S.: Context-sensitive dependency pairs. In: FSTTCS'06. Volume 4337 of LNCS. (2006) 298–309
9. Hirokawa, N., Middeldorp, A.: Dependency pairs revisited. In: RTA 2004. Volume 3091 of LNCS. (2004) 249–268
10. Termination Competition: `www.lri.fr/~marche/termination-competition/`
11. Giesl, J., Thiemann, R., Schneider-Kamp, P.: Proving and disproving termination of higher-order functions. In: FroCoS 2005. Volume 3717 of LNCS. (2005) 216–231

[4] We did not employ Theorem 3.4, as choosing a zero matrix for an argument has the same effect as filtering on that argument.