Termination Dependencies

Nachum Dershowitz*
School of Computer Science, Tel Aviv University, Israel
Email: nachumd@tau.ac.il

The innovative *dependency-pair* termination method of [1, 2] relies on two important observations:

- If a rewrite system is nonterminating, then there is an infinite derivation with at least one redex at the top of a term (see, for example, [4, p. 287]).
- If a rewrite system is nonterminating, then there is an infinite derivation in which all proper subterms of every redex are *mortal* (these are the "constricting" derivations of [11]). By "mortal", we mean that it initiates finite derivations only.

Let F be some vocabulary (of constant and function symbols) and T, the set of terms constructed from it. The dependency-pair method can be reformulated—and somewhat strengthened—in terms of two (related) quasi-orderings, as follows:

> *A rewrite system terminates if there are well-founded quasi-orderings* $\succsim$ *and* $\succsim'$ *such that*:
> 1. **(Rule)** $\ell \succsim r$ for all rules $\ell \to r$;
> 2. **(Dependency)** $\ell \succ' u$ for all subterms u of the right side r of a rule $\ell \to r$ that are not also subterms of the left side ℓ;
> 3. **(Monotonicity)** $u \succsim v$ implies $f(\cdots,u,\cdots) \succsim f(\cdots,v,\cdots)$ for all symbols $f \in F$ and (ground) terms $\cdots,u,v,\cdots \in T$; and
> 4. **(Harmony)** $u \succsim v$ implies $f(\cdots,u,\cdots) \succsim' f(\cdots,v,\cdots)$ for all symbols $f \in F$ and (ground) terms $\cdots,u,v,\cdots \in T$,
>
> *where* $\succ$ *and* $\succ'$ *are the strict partial orderings (*$\succsim \setminus \precsim$ *and* $\succsim' \setminus \precsim'$*) associated with* $\succsim$ *and* $\succsim'$.

Only $\succsim$ is required to be monotonic. We have excluded dependency inequalities for right-hand subterms that also appear on the left; this includes all variables. Harmony is called "quasi-monotonicity of $\succsim'$ with respect to $\succsim$" in [3]. (All inequalities refer to ground terms but can be lifted to free terms in the standard manner: Demanding that $u \succ v$ for terms u,v with free variables means that $u\gamma \succ v\gamma$ for all substitutions γ of ground terms for those variables; see also [2, fn. 5].)

To clarify relations between various termination methods, we first define some properties of binary term relations:

$$\begin{aligned}
\text{Sub}(\sqsupset): &\quad f(\cdots s\cdots) \sqsupset s\\
\text{Mono}(\sqsupset): &\quad s \sqsupset t \Rightarrow f(\cdots s\cdots) \sqsupset f(\cdots t\cdots)\\
\text{Harmony}(\sqsupset,\gg): &\quad s \sqsupset t \Rightarrow f(\cdots s\cdots) \gg f(\cdots t\cdots)\\
\text{Compat}(\sqsupset,\gg): &\quad s \sqsupset t \gg u \Rightarrow s \sqsupset u\\
\text{Rule}_R(\sqsupset): &\quad \ell \to r \in R \Rightarrow \ell \sqsupset r\\
\text{Reduce}_R(\sqsupset): &\quad s \to_R t \Rightarrow s \sqsupset t\\
\text{Depend}_R(\sqsupset): &\quad \ell \to r \in R \wedge s \trianglelefteq r \Rightarrow \ell \sqsupset s \vee s \lhd \ell
\end{aligned}$$

* This research was supported in part by The Israel Science Foundation (grant no. 254/01).

where R is an arbitrary rewrite system; $\to_R$ is the associated rewrite relation; and $\trianglelefteq$ ($\triangleleft$) is the (proper) subterm relation. All symbols should be understood universally: s,t,u are arbitrary ground terms; ellipses $\cdots$ represent arbitrary lists of terms; $\sqsupset, \gg$ are arbitrary binary relations over terms; and f is an arbitrary function symbol (or constant).

In what follows, let $\succsim$ and $\succsim'$ be arbitrary *well-founded* quasi-orderings. Using the above notation, we can express our version (somewhat stronger that the suggestion in [6, p. 558, *en passant*]) of the dependency method as follows:

Dependency ($\succsim, \succsim'$) [6]**:** $\text{Depend}_R(\succ')$, $\text{Rule}_R(\succsim)$, $\text{Mono}(\succsim)$, $\text{Harmony}(\succsim,\succsim')$.

More precisely this means

$$\begin{array}{l}\text{WFO}(\succsim) \wedge \text{WFO}(\succsim') \wedge \\ \text{Depend}_R(\succ') \wedge \text{Rule}_R(\succsim) \wedge \text{Mono}(\succsim) \wedge \text{Harmony}(\succsim, \succsim')\end{array} \Rightarrow \text{SN}(R)$$

where

$$\begin{array}{rl}\text{WF}(\gg): & \text{no infinite descending sequences } x_1 \gg x_2 \gg \cdots \\ \text{WFO}(\succsim): & \succsim \text{ is a quasi-ordering and WF}(\succ) \\ \text{SN}(R): & \text{WF}(\to_R)\end{array}$$

Let $\widehat{F}$ be a mirror image of F: $\widehat{F} = \{\widehat{g} | g \in F\}$. Denote by $\widehat{s}$ the term $s = f(u_1,\ldots,u_n)$ with root symbol $f \in F$ replaced by its mirror image $\widehat{f} \in \widehat{F}$, that is, $\widehat{s} = \widehat{f}(u_1,\ldots,u_n)$. Let $\widehat{T}$ be T's image under $\widehat{\cdot}$. If $\succ$ is a partial ordering of $\widehat{T}$, define another partial ordering $\widehat{\succ}$ as $u \mathrel{\widehat{\succ}} v$, for terms $u, v \in T$, when $\widehat{u} \succ \widehat{v}$. The original method of [2] is approximately:

Dependency Pairs ($\succsim$) [2]**:** $\text{Depend}_R(\widehat{\succ})$, $\text{Rule}_R(\succsim)$, $\text{Mono}(\succsim)$.

where Mono applies to both hatted ($f \in \widehat{F}$) and bareheaded ($f \in F$) terms. A more recent version [8] of the dependency-pair method is essentially:

Variant ($\succsim, \succ'$) [8]**:** $\text{Depend}_R(\succ')$, $\text{Rule}_R(\succsim)$, $\text{Mono}(\succsim)$, $\text{Compat}(\succ', \succsim)$.

The methods of [2, 6, 8] exclude only variable subterms x of r from the requirement that $\ell \mathrel{\widehat{\succ}} x$ or $\ell \succ' x$, but the proofs remain valid for all $s \triangleleft \ell$, as in our condition $\text{Depend}_R(\succ')$. There is no need to explicitly exclude the case that s is headed by a constructor (as done in [2]). Instead, one simply makes all terms headed by a constructor smaller under $\succ$ than terms headed by defined symbols (doable, since it need not be monotonic). In fact, any term that can never have a top redex, regardless of rewrites below the top, can be made minimal, and be safely ignored in the same way.

The dependency-pair method is derived in [2] from the condition:

Main ($\succsim$) [2]**:** $\text{Depend}_R(\succ)$, $\text{Mono}(\to_R \cap \succsim)$.

We contribute the following variants:

Basic ($\succsim$)**:** $\text{Depend}_R(\succ)$, $\text{Reduce}_R(\succsim)$.

Intermediate ($\succsim, \succ'$)**:** $\text{Depend}_R(\succ')$, $\text{Reduce}_R(\succsim)$, $\text{Compat}(\succ', \succsim)$.

The following summarizes dependencies between the different methods:

Lemma 1.

1. **Main** ($\succsim$) $\Rightarrow$ **Basic** ($\succsim$)
2. **Basic** ($\succsim^*$) $\Rightarrow$ **Intermediate** ($\succsim$,$\succ'$)
3. **Intermediate** ($\succsim$,$\succ'$) $\Rightarrow$ **Variant** ($\succsim$,$\succ'$)
4. **Main** ($\succsim'$) $\Rightarrow$ **Dependency** ($\succsim$,$\succsim'$)
5. **Dependency** ($\succsim$,$\widehat{\succsim}$) $\Rightarrow$ **Dependency Pairs** ($\succsim$)

where $\succsim^*$ is the transitive closure of $\succsim \cup \succsim'$ (which is well-founded when the two are compatible).

An implication $M \Rightarrow M'$ means that method M' is a special case of method M. To prove the implication, viz. that correctness of the antecedent method M implies correctness of the consequent M', one shows that the *requirements* for M' imply the requirements for M. Thus, any well-founded ordering(s) used by M should be derivatives of those used by M'.

For example, to prove Lemma 1.2, we need to show

$$\begin{aligned} \mathrm{WFO}(\succsim) \wedge \mathrm{WFO}(\succsim') \wedge \mathrm{Depend}_R(\succ') \wedge \mathrm{Reduce}_R(\succsim) \wedge \mathrm{Compat}(\succ',\succsim) \\ \Rightarrow \quad \mathrm{WFO}(\succsim^*) \wedge \mathrm{Depend}_R(\succ^*) \wedge \mathrm{Reduce}_R(\succsim^*) \end{aligned}$$

which follows from compatibility and properties of the transitive closures.

Other termination methods can be summarized using these same properties:

Standard ($\succ$) [10]**:** $\mathrm{Rule}_R(\succ)$, $\mathrm{Mono}(\succ)$.

Kamin & Lévy ($\succ$) [9]**:** $\mathrm{Rule}_R(\succ)$, $\mathrm{Mono}(\rightarrow_R \cap \succ)$.

Quasi-Simplification Ordering ($\succsim$) [4]**:** $\mathrm{Rule}_R(\succ)$, $\mathrm{Sub}(\succsim)$, $\mathrm{Mono}(\succsim)$.

Subterm ($\succsim$) [5]**:** $\mathrm{Rule}_R(\succ)$, $\mathrm{Sub}(\succsim)$, $\mathrm{Mono}(\rightarrow_R \cap \succsim)$.

We can show:

Lemma 2.

1. **Standard** ($\rightarrow_R \cap \succ$) $\Rightarrow$ **Kamin & Lévy** ($\succ$)
2. **Main** ($\succsim$) $\Rightarrow$ **Subterm** ($\succsim$)
3. **Kamin & Lévy** ($\succ^\omega$) $\Rightarrow$ **Subterm** ($\succsim$)
4. **Subterm** ($\succsim$) $\Rightarrow$ **Quasi-Simplification Ordering** ($\succsim$)

where $s \succ^\omega t$ is the well-founded multiset extension [7] of $\succ$ to the bag of all subterms of s,t.

One application of the weakened method shown here could be a "pattern-based" ordering. For the non-monotonic "surface" ordering, one can check whether the term matches a given pattern derived from a rule's left side.

References

1. Thomas Arts. *Automatically Proving Termination and Innermost Normalization of Term Rewriting Systems*. PhD thesis, Universiteit Utrecht, Utrecht, The Netherlands, 1997.
2. Thomas Arts and Jürgen Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236:133–178, 2000.
3. Cristina Borralleras, Maria Ferreira and Albert Rubio. Complete monotonic semantic path orderings. In: David A. McAllester, ed., *Proceedings of the 17th International Conference on Automated Deduction (CADE)*, *Lecture Notes in Computer Science* 1831, Pittsburgh, PA, Springer Verlag, pages 346–364, June 2000.
4. Nachum Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, March 1982.
5. Nachum Dershowitz. Termination of rewriting. *J. Symbolic Computation*, 3(1&2):69–115, February/April 1987.
6. Nachum Dershowitz and David A. Plaisted. Rewriting. Chap. 9 in: *Handbook of Automated Reasoning*, vol. 1, A. Robinson and A. Voronkov, eds., Elsevier Science, pp. 535–610, 2001.
7. Nachum Dershowitz and Zohar Manna. Proving termination with multiset orderings. *Communications of the ACM*, 22(8):465–476, August 1979.
8. Jürgen Giesl and Deepak Kapur. Dependency pairs for equational rewriting. In: *Proceedings of the 12th International Conference on Rewriting Techniques and Applications (RTA-01)*, Utrecht, The Netherlands, *Lecture Notes in Computer Science* 2051, Springer Verlag, pages 93–107, 2001.
9. Sam Kamin and Jean-Jacques Lévy. Two generalizations of the recursive path ordering. Unpublished note, Department of Computer Science, University of Illinois, Urbana, IL, February 1980.
10. Dallas S. Lankford. On proving term rewriting systems are Noetherian. Memo MTP-3, Mathematics Department, Louisiana Tech. University, Ruston, LA, May 1979. Revised October 1979.
11. David A. Plaisted. Semantic confluence tests and completion methods. *Information and Control*, 65(2/3):182–215, 1985.