33 Examples of Termination*

Nachum Dershowitz

Department of Computer Science, University of Illinois, Urbana, IL 61801, USA

Abstract. A graded sequence of examples—presented in a uniform framework—spotlights stages in the development of methods for proving termination of rewrite systems.

Let $\mathcal{T}$ be the set of all terms over some vocabulary. A *rewrite system* over $\mathcal{T}$ is a (finite or infinite) set of *rules*, each of the form $l \rightarrow r$, where l and r are terms containing variables ranging over $\mathcal{T}$. A rule $l \rightarrow r$ applies to a term t in $\mathcal{T}$ if a subterm s of t matches the left-hand side l with some substitution σ of terms in $\mathcal{T}$ for variables appearing in l. The rule is applied by replacing the *redex* s in t with the corresponding right-hand side $r\sigma$ of the rule, to which the same substitution σ of terms for variables has been applied. We write $t \rightarrow u$ to indicate that the term t in $\mathcal{T}$ *rewrites* in this way to the term u in $\mathcal{T}$ by a single application of some rule. Note that more than one rule can apply to t and rules can apply at more than one subterm s. Rewrite systems have long been used as decision procedures for validity in equational theories, that is, for truth of an equation in all models of the theory. They are also used as a specification and programming language. See [Dershowitz and Jouannaud, 1990; Klop, 1992; Plaisted, 1993] for recent surveys of term-rewriting and some of its applications.

A rewrite system is *terminating* if there are no infinite derivations $t_1 \rightarrow t_2 \rightarrow t_3 \cdots$. Termination is undecidable. A proof of termination must take into consideration the many different possible rewrite sequences permitted by the *nondeterministic* choice of rules and subterms. We present a series of examples culled from the literature to illustrate the progression of techniques used to prove termination of vanilla-flavored rewriting.[1] We describe these techniques in a manner designed to highlight similarities and follow a logical sequence that is not perfectly chronological. Not covered here are methods based on transformations of the given system, including those for more intricate forms of rewriting (notably, when permutation of operands of associative-commutative operators is permitted prior to rewriting). A survey of termination methods for rewriting may be found in [Dershowitz, 1987].

Example 1 (Loops). T. Evans [1951] gave the first rewrite-based decision procedure, using these dozen rules to compute normal forms:

$$
\begin{array}{rclcrcl}
x \backslash x & \rightarrow & e & \quad & x \cdot (x \backslash y) & \rightarrow & y \\
x / x & \rightarrow & e & \quad & (y/x) \cdot x & \rightarrow & y \\
e \cdot x & \rightarrow & x & \quad & x \backslash (x \cdot y) & \rightarrow & y \\
x \cdot e & \rightarrow & x & \quad & (y \cdot x) \backslash x & \rightarrow & y \\
e \backslash x & \rightarrow & x & \quad & x / (y \backslash x) & \rightarrow & y \\
x / e & \rightarrow & x & \quad & (x/y) \backslash x & \rightarrow & y \,.
\end{array}
$$

This system obviously terminates, since application of any rule to any redex in t decreases the size (number of symbols) $|t|$ of t. To quote Evans, "[T]he effect of an elementary reduction is to reduce the length of a word by at least 1." □

Example 2 (Fragment of Group Theory). Similarly,

$$
\begin{array}{rclcrcl}
1 \cdot x & \rightarrow & x & \quad & x \cdot 1 & \rightarrow & x \\
x^- \cdot x & \rightarrow & 1 & \quad & x \cdot x^- & \rightarrow & 1 \\
1^- & \rightarrow & 1 & \quad & (x^-)^- & \rightarrow & x \\
x^- \cdot (x \cdot y) & \rightarrow & y & \quad & x \cdot (x^- \cdot y) & \rightarrow & y
\end{array}
$$

[1] I apologize for not providing the provenance of many of the examples.

terminates, since

$$s \to t \Rightarrow |s| > |t| \ .$$

Note that it is not enough for every rule's left-hand side to be longer than its right-hand side. If the right-hand side has more occurrences of some variable than does the left, as in a rule like $(x^-) \cdot y \to y \cdot y$, then an application like $a^- \cdot (a \cdot a) \to (a \cdot a) \cdot (a \cdot a)$ causes an increase in size.

The size (length) ordering has been particularly popular in work on semigroups, for which $|l| > |r|$, for each rule $l \to r$, suffices to ensure termination. □

Example 3. Clearly, measures other than size can be used. For example,

$$f(f(x)) \quad \to \quad g(f(x))$$

terminates, since the number of f's decreases. □

Example 4. The lexicographic combination of well-founded orderings is well-founded, so one can show termination of

$$
\begin{aligned}
f(f(x)) &\quad \to \quad g(f(x)) \\
g(g(x)) &\quad \to \quad f(x)
\end{aligned}
$$

by considering the pair $\langle \mathbf{size}, \mathbf{number\ of\ } f\mathbf{s} \rangle$. □

Example 5. In general, one can use any measure $[\![\cdot]\!] : \mathcal{T} \to W$ such that

$$s \to t \Rightarrow [\![s]\!] \succ [\![t]\!] \ ,$$

where $\succ$ is a well-founded ordering on W. For example,

$$f(f(x)) \quad \to \quad f(g(f(x)))$$

terminates, since the number of adjacent f's decreases, regardless of what is substituted for x, or what context surrounds $f(f(x))$. □

Example 6. To show termination of

$$
\begin{aligned}
f(g(x)) &\quad \to \quad g(g(f(x))) \\
f(g(x)) &\quad \to \quad g(g(g(x))) \ ,
\end{aligned}
$$

we can use the suggestion of S. Gorn [1973] and consider the tuple of heights (distance from constants in the tree representation of the term) of the fs. Moving an f right decreases the height of that f while increasing the height of fs to its left, so we compare the heights lexicographically: $\langle \mathbf{number\ of\ } f\mathbf{s}, \mathbf{height\ of\ rightmost\ } f, \ldots, \mathbf{height\ of\ leftmost\ } f \rangle$. □

Example 7 (Dutch National Flag). Consider the rules:

$$
\begin{aligned}
w(r(x)) &\quad \to \quad r(w(x)) \\
b(r(x)) &\quad \to \quad r(b(x)) \\
b(w(x)) &\quad \to \quad w(b(x)) \ .
\end{aligned}
$$

The term can be viewed as an integer in base 3, with "blue" $b = 2$, "white" $w = 1$, and "red" r and any other symbol appearing in a term as 0. □

Example 8 (Differentiation). One of the problems on a qualifying exam given by R. Floyd at Carnegie-Mellon University in 1967 was to prove termination of

$$
\begin{aligned}
D\mathbf{t} &\rightarrow 1 \\
D(\text{constant}) &\rightarrow 0 \\
D(x + y) &\rightarrow Dx + Dy \\
D(x \times y) &\rightarrow (y \times Dx) + (x \times Dy) \\
D(x - y) &\rightarrow Dx - Dy \ .
\end{aligned}
$$

He had in mind a proof using ordinals, such as

$$
\begin{aligned}
[\![Dx]\!] = \omega^{[\![x]\!]} &\qquad [\![x + y]\!] = [\![x]\!] + [\![y]\!] \\
[\![\mathbf{t}]\!] = 1 &\qquad [\![x - y]\!] = [\![x]\!] + [\![y]\!] \\
[\![\text{constant}]\!] = 1 &\qquad [\![x \times y]\!] = [\![x]\!] + [\![y]\!] \ ,
\end{aligned}
$$

where the sums and products are commutative. □

Example 9 (Semigroups). Z. Manna (who took Floyd's exam) and S. Ness (Manna's student) [1970] suggested a general method based on interpretations $[\![\cdot]\!]$: assign an n-ary function $[\![f]\!] : W^n \rightarrow W$, for some well-founded set W, to each n-ary symbol f, and let $[\![\cdot]\!]$ interpret terms accordingly. With the added *replacement property* (true of the above ordinal notations),

$$
x > y \quad \Rightarrow \quad [\![f]\!](\ldots x \ldots) > [\![f]\!](\ldots y \ldots) \ ,
$$

for each symbol f, one need only show

$$
\forall \bar{x} \ [\![l]\!] > [\![r]\!] \ ,
$$

where $\bar{x}$ are the variables appearing in l and r. For

$$
(x \cdot y) \cdot z \quad \rightarrow \quad x \cdot (y \cdot z) \ ,
$$

take

$$
[\![x \cdot y]\!] = 2[\![x]\!] + [\![y]\!] \qquad [\![\text{constant}]\!] = 2 \ .
$$

It can be shown that termination of a system for one vocabulary (as here for $\cdot$ and constants) implies termination over any larger vocabulary (which adds symbols not appearing in the system's rules). □

Example 10. Another student taking that test, R. Iturriaga, went on to produce a dissertation on symbolic computation [Iturriaga, 1967] in which a class of systems, including differentiation, were proved terminating. The idea was to use exponential interpretations; the following is somewhat simplified:

$$
\begin{aligned}
[\![x + y]\!] = [\![x]\!] + [\![y]\!] &\qquad\qquad [\![Dx]\!] = 3^{[\![x]\!]} \\
[\![x - y]\!] = [\![x]\!] + [\![y]\!] &\qquad\qquad [\![\mathbf{t}]\!] = 3 \\
[\![x \times y]\!] = [\![x]\!] + [\![y]\!] &\qquad [\![\text{constant}]\!] = 3
\end{aligned}
$$

□

Example 11. D. Lankford [1979] championed the use of polynomial interpretations, which suffice even for an expanded differentiation program:

$$
\begin{aligned}
D\mathbf{t} &\rightarrow 1 \\
D(\text{constant}) &\rightarrow 0 \\
D(x + y) &\rightarrow Dx + Dy \\
D(x \times y) &\rightarrow (y \times Dx) + (x \times Dy) \\
D(x - y) &\rightarrow Dx - Dy \\
D(-x) &\rightarrow -Dx \\
D(x/y) &\rightarrow (Dx/y) - (x \times Dy/y^2) \\
D(\ln x) &\rightarrow Dx/x \\
D(x^y) &\rightarrow (y \times x^{y-1} \times Dx) + (x^y \times (\ln x) \times Dy) \ .
\end{aligned}
$$

Let

$$\begin{aligned}
[\![x + y]\!] &= [\![x]\!] + [\![y]\!] & [\![x \times y]\!] &= [\![x]\!] + [\![y]\!] \\
[\![x - y]\!] &= [\![x]\!] + [\![y]\!] & [\![x/y]\!] &= [\![x]\!] + [\![y]\!] \\
[\![x^y]\!] &= [\![x]\!] + [\![y]\!] & [\![Dx]\!] &= [\![x]\!]^2 \\
[\![-x]\!] &= [\![x]\!] + 1 & [\![\ln x]\!] &= [\![x]\!] + 1 \\
[\![\text{constant}]\!] &= 2 & [\![\mathbf{t}]\!] &= 2 \;.
\end{aligned}$$

$\square$

Example 12 (Disjunctive Normal Form). For

$$\begin{aligned}
\neg\neg x &\rightarrow x \\
\neg(x \vee y) &\rightarrow (\neg x) \wedge (\neg y) \\
\neg(x \wedge y) &\rightarrow (\neg x) \vee (\neg y) \\
x \wedge (y \vee z) &\rightarrow (x \wedge y) \vee (x \wedge z) \\
(y \vee z) \wedge x &\rightarrow (x \wedge y) \vee (x \wedge z) \;,
\end{aligned}$$

exponentials are, however, required:

$$\begin{aligned}
[\![x \vee y]\!] &= [\![x]\!] + [\![y]\!] + 1 & [\![\neg x]\!] &= 2^{[\![x]\!]} \\
[\![x \wedge y]\!] &= [\![x]\!][\![y]\!] & [\![\text{constant}]\!] &= 2 \;.
\end{aligned}$$

Primitive recursive interpretations cannot prove termination of all terminating systems, so resort to ordinals or their equivalents is inevitable. $\square$

Example 13. Sometimes, pairs of interpretations come in handy: For

$$\begin{aligned}
x \times (y + z) &\rightarrow (x \times y) + (x \times z) \\
(y + z) \times x &\rightarrow (x \times y) + (x \times z) \\
(x \times y) \times z &\rightarrow x \times (y \times z) \\
(x + y) + z &\rightarrow x + (y + z) \;,
\end{aligned}$$

we can use $\langle [\![\mathbf{term}]\!], [\![\mathbf{term}]\!]' \rangle$, where

$$\begin{aligned}
[\![x \times y]\!] &= [\![x]\!][\![y]\!] & [\![x \times y]\!]' &= 2[\![x]\!]' + [\![y]\!]' \\
[\![x + y]\!] &= [\![x]\!] + [\![y]\!] + 1 & [\![x + y]\!]' &= 2[\![x]\!]' + [\![y]\!]' \\
[\![\text{constant}]\!] &= 2 & [\![\text{constant}]\!]' &= 2 \;.
\end{aligned}$$

The first two rules decrease the first interpretation; the last two decrease the second, without changing the first. So, a lexicographic comparison of pairs is in order. $\square$

Example 14. Other times, it pays to map terms to pairs [Zantema, 1991]. Looking again at

$$f(f(x)) \rightarrow f(g(f(x))) \;,$$

we can let $[\![\cdot]\!] : \mathcal{T} \rightarrow N \times N$ as follows: $[\![f]\!]\langle x, y \rangle = \langle x + y, x \rangle$, $[\![g]\!]\langle x, y \rangle = \langle y, x \rangle$, and $[\![\text{constants}]\!] = \langle 1, 1 \rangle$. Since the interpretation of the term being rewritten decreases in its second component, while the first remains intact, the whole term's value can also be shown decreasing. $\square$

Example 15. Another way to show termination of

$$\begin{aligned}
f(g(x)) &\rightarrow g(g(f(x))) \\
f(g(x)) &\rightarrow g(g(g(x)))
\end{aligned}$$

is to look at the pair $\langle \mathbf{top}, \mathbf{argument} \rangle$, where the top (leftmost) symbols are compared in a precedence (ordering of function symbols), with $f > g$. Both rules decrease the top symbol, and the second component ensures that the decrease carries over recursively to the whole term. For such recursive comparisons to yield a well-founded ordering, the second component must be bounded. That is, one must show that the recursive component (the suffix of the right-hand side) is smaller than the whole left-hand side. For this to in fact be the case, we need to include the proper subterm relation in the ordering, that is, we must have $f(x), g(x) \succ x$. This is the simplest example of a "path ordering" [Plaisted, 1978a; Dershowitz, 1982].

Example 16. Consider, again,

$$(x \cdot y) \cdot z \quad \rightarrow \quad x \cdot (y \cdot z)$$

and the triple ⟨**size, first multiplicand, second multiplicand**⟩, where the multiplicands are compared recursively in this lexicographic ordering (for a constant term, minimal elements can be used instead). As before, we need to check that the left-hand side is strictly greater than all the subterms of the right-hand side. □

Example 17 (Group Theory). The first decision procedure obtained by D. Knuth and P. Bendix's [1970] completion program was the following system:

$$
\begin{array}{rclcrcl}
1 \cdot x & \rightarrow & x & \qquad & x \cdot 1 & \rightarrow & x \\
x^- \cdot x & \rightarrow & 1 & & x \cdot x^- & \rightarrow & 1 \\
1^- & \rightarrow & 1 & & x^{--} & \rightarrow & x \\
y^- \cdot (y \cdot z) & \rightarrow & z & & y \cdot (y^- \cdot z) & \rightarrow & z \\
(x \cdot y) \cdot z & \rightarrow & x \cdot (y \cdot z) & & (x \cdot y)^- & \rightarrow & y^- \cdot x^- \ .
\end{array}
$$

To prove its termination, Knuth devised a recursive ordering that combined the notion of precedence with a simple linear weight: ⟨**weight, top, first argument, . . . , last argument**⟩. (The arguments can be listed in any other permutation just as well.) The weight is the sum of the weights of all the symbols, which are non-negative integers. Symbols are compared in the precedence order; arguments are compared recursively. Constants must have positive weight and a unary symbol may have zero weight only if it is maximal in the precedence. This ensures that terms are greater than their subterms, and, hence, that the arguments are bounded. For the group example, give constants weight 1, other operators weight 0, and let inverse have the greatest precedence. Weight is needed for the simple rules; the precedence is what makes the rule for distributing inverse work; the lexicographic comparison of subterms is all that is needed for associativity. □

Example 18 (Distributivity). For

$$x \times (y + z) \quad \rightarrow \quad (x \times y) + (x \times z) \ ,$$

R. Lipton and L. Snyder [1977] suggested the pair ⟨**natural interpretation, maximum - size**⟩. The "natural" interpretation,

$$
\begin{aligned}
[\![x \times y]\!] &= [\![x]\!][\![y]\!] \\
[\![x + y]\!] &= [\![x]\!] + [\![y]\!] \qquad [\![\text{constant}]\!] = 1 \ ,
\end{aligned}
$$

is unchanged by rewriting, hence imposes a maximum size (and height) on equal terms, which is then used in the second component to show a decrease. □

Example 19. As Knuth and Bendix pointed out, their method needs to be extended to handle "duplicating" systems (one that has more occurrences of a variable on the right than on the left), such as:

$$
\begin{array}{rcl}
x \times (y + z) & \rightarrow & (x \times y) + (x \times z) \\
(y + z) \times x & \rightarrow & (x \times y) + (x \times z) \\
(x \times y) \times z & \rightarrow & x \times (y \times z) \\
(x + y) + z & \rightarrow & x + (y + z) \ .
\end{array}
$$

D. Lankford [1979] suggested extending the method of Knuth and Bendix by using integer polynomial weights (with positive coefficients to guarantee that terms are greater than subterms): ⟨$[\![$**term**$]\!]$**, top, left argument, right argument**⟩. A natural interpretation, with precedence $\times > +$ does the trick. □

Example 20. To prove termination of

$$\begin{aligned}
\neg\neg x &\;\rightarrow\; x \\
\neg(x \vee y) &\;\rightarrow\; (\neg\neg\neg x) \wedge (\neg\neg\neg y) \\
\neg(x \wedge y) &\;\rightarrow\; (\neg\neg\neg x) \vee (\neg\neg\neg y) \;,
\end{aligned}$$

use the multiset {number of and's and or's in $x | \neg x$ in term}. Finite multisets are compared using the well-founded multiset ordering [Dershowitz and Manna, 1979] in which replacing an element with any number of smaller elements decreases the multiset. Since applying a rule does not change the total number of and's and or's, the contribution of superterms of the redex is unchanged thereby. □

Example 21 (Factorial). Oftentimes, one would like to consider the top symbol before the interpretation. That way, one can ignore less significant functions. For

$$\begin{aligned}
p(s(x)) &\;\rightarrow\; x \\
fact(0) &\;\rightarrow\; s(0) \\
fact(s(x)) &\;\rightarrow\; s(x) \times fact(p(s(x))) \\
0 \times y &\;\rightarrow\; 0 \\
s(x) \times y &\;\rightarrow\; (x \times y) + y \\
x + 0 &\;\rightarrow\; x \\
x + s(y) &\;\rightarrow\; s(x + y) \;,
\end{aligned}$$

we would use a precedence $fact > \times > + > s > 0$. Then we compare two calls to $fact$ by their natural interpretation:

$$\begin{aligned}
[\![fact(x)]\!] = [\![x]\!]! & \qquad [\![x \times y]\!] = [\![x]\!][\![y]\!] \\
[\![s(x)]\!] = [\![x]\!] + 1 & \qquad [\![x + y]\!] = [\![x]\!] + [\![y]\!] \\
[\![p(x)]\!] = [\![x]\!] - 1 & \qquad [\![0]\!] = 0 \;.
\end{aligned}$$

To compare terms, we consider $\langle \mathbf{top}, [\![\mathbf{term}]\!] \rangle$, and also let terms be greater than subterms. Instead of the replacement property, one need only ensure that rewriting a subterm does not increase the whole term:

$$s \succ t \text{ and } s \rightarrow t \quad\Rightarrow\quad g(\ldots, s, \ldots) \succeq g(\ldots, t, \ldots) \;,$$

for all symbols g, which is not a problem since the interpretation is "value-preserving."

   It is enough to show that superterms (as opposed to the redex) do not increase (rather than actually decrease), since this implies that the multiset of pairs for all subterms shows a strict decrease, on account of the fact that terms are taken to be larger than subterms [Dershowitz, 1982]. □

Example 22. Another approach, based on multisets, for proving termination of differentiation is D. Plaisted's [1978a] *simple path ordering*. Terms are mapped into multisets of sequences of function symbols:

$$[\![t]\!] \;=\; \{\text{paths in } t\} \;,$$

where a *path* is a sequence of function symbols, starting with the root symbol, and taking subterms until a constant is reached. Sequences are compared as in Example 15, with the differentiation operator maximal in the precedence. □

Example 23. Nested multiset structures can also be used to prove termination of the differentiation example [Dershowitz and Manna, 1979]:

$$\begin{aligned}
[\![Dx]\!] = \{[\![x]\!]\} & \qquad [\![x + y]\!] = [\![x]\!] \cup [\![y]\!] \\
[\![-x]\!] = [\![x]\!] \cup \{\emptyset\} & \qquad [\![x - y]\!] = [\![x]\!] \cup [\![y]\!] \\
[\![\ln x]\!] = [\![x]\!] \cup \{\emptyset\} & \qquad [\![x \times y]\!] = [\![x]\!] \cup [\![y]\!] \\
[\![x/y]\!] = [\![x]\!] \cup [\![y]\!] & \qquad [\![\text{constant}]\!] = \{\emptyset\} \\
[\![x^y]\!] = [\![x]\!] \cup [\![y]\!] & \qquad [\![\mathbf{t}]\!] = \{\emptyset\} \;.
\end{aligned}$$

This is really just an encoding of Floyd's ordinal-based solution. □

Example 24. Consider, once again,

$$(x \cdot y) \cdot z \quad \rightarrow \quad x \cdot (y \cdot z) \ .$$

All we really want to consider is the size of the first multiplicand. There could, however, be a product in z with a multiplicand larger than $x \cdot y$, so one needs to add another factor: $\langle \mathbf{size}, \mathbf{size\ of\ first\ multiplicand} \rangle$, which is simpler than a Knuth-Bendix ordering. Since rewriting does not affect the size of superterms, the whole term does not increase.

Better yet, one can use $\langle \mathbf{size\ of\ first\ multiplicand} \rangle$, and say that $s \succ t$ if t is a proper subterm of s or if they're both products, but s's first multiplicand is longer. The ordering is, thus, the transitive closure of the proper subterm relation and the ordering based on the size of the first multiplicand. □

Example 25. For the factorial system, one can imitate a structural-induction proof of termination of the corresponding recursive definitions using $\langle \mathbf{top}, [\![\mathbf{argument}]\!] \rangle$, where the precedence is as before, $fact > \times > + > s > 0$, simplifying the interpretation to

$$\begin{aligned}
[\![fact(x)]\!] &= 0 & [\![x \times y]\!] &= [\![x]\!][\![y]\!] \\
[\![s(x)]\!] &= [\![x]\!] + 1 & [\![x + y]\!] &= [\![x]\!] + [\![y]\!] \\
[\![p(x)]\!] &= [\![x]\!] - 1 & [\![0]\!] &= 0 \ ,
\end{aligned}$$

and taking the first argument of factorial and times, and the second for plus. Again, this is combined with the subterm relation. The value-preserving nature of the interpretation guarantees that superterms do not increase. □

Example 26. The disjunctive normal form example served as motivation for the development of the *recursive path ordering* in [Dershowitz, 1982], in parallel with related ideas in [Plaisted, 1978b]:

$$\begin{aligned}
\neg\neg x &\rightarrow x \\
\neg(x \vee y) &\rightarrow (\neg x) \wedge (\neg y) \\
\neg(x \wedge y) &\rightarrow (\neg x) \vee (\neg y) \\
x \wedge (y \vee z) &\rightarrow (x \wedge y) \vee (x \wedge z) \\
(y \vee z) \wedge x &\rightarrow (x \wedge y) \vee (x \wedge z) \ .
\end{aligned}$$

Intuitively, the precedence should be $\neg > \wedge > \vee$, and terms should be greater than subterms. Adding the subterms as a second component: $\langle \mathbf{top}, \{\mathbf{arguments}\} \rangle$, we need to ascertain that the left-hand sides are greater in this ordering than all subterms of the corresponding right-hand sides. We use multisets in this path ordering, so that $(y \vee z) \wedge x \succ (x \wedge y)$. □

Example 27. As with previous methods, sometimes two orderings are better than one. For example, the first rule of

$$\begin{aligned}
h(f(x), y) &\rightarrow f(g(x, y)) \\
g(x, y) &\rightarrow h(x, y)
\end{aligned}$$

suggests the precedence $h \approx g > f$; the second rule, on the other hand, requires $g > h$. So we prove termination with $\langle \mathbf{term}, \mathbf{number\ of\ } g\text{'s} \rangle$, using a multiset path ordering based on $h \approx g > f$ for the first component (under which two terms are equivalent when symbols are replaced with equivalents and/or arguments are permuted). □

Example 28 (Conditional Normalization). To prove termination of

$$if(if(x, y, z), u, v) \quad \rightarrow \quad if(x, if(y, u, v), if(z, u, v)) \ ,$$

one can treat the first argument of *if* as the top symbol in a recursive path ordering [Dershowitz, 1982]: $\langle \mathbf{first\ argument}, \{\mathbf{other\ arguments}\} \rangle$. Rather than an ordinary precedence, one compares first arguments recursively. □

Example 29 (Ackermann's Function). S. Kamin and J.-J. Levy [1980] suggested a lexicographic version of the recursive path ordering, which is like Knuth-Bendix, but with all terms having the same weight: $\langle \mathbf{top}, \mathbf{first\ argument}, \dots, \mathbf{last\ argument}\rangle$, and terms greater than subterms. This works beautifully with examples like:

$$
\begin{aligned}
ack(0, y) &\rightarrow succ(y) \\
ack(succ(x), 0) &\rightarrow ack(x, succ(0)) \\
ack(succ(x), succ(y)) &\rightarrow ack(x, ack(succ(x), y)) \,,
\end{aligned}
$$

where the precedence $ack > succ$ is used for the first rule, and the lexicographic aspect for the others. To establish that $s \succ f(t_1, \dots, t_n)$ in this ordering, one must also check (recursively) that $s \succ t_1, \dots, t_n$, i.e. that $ack(succ(x), succ(y)) \succ ack(succ(x), y)$. □

Example 30 (Combinator C). With this lexicographic path ordering, the following can be shown to terminate with no precedence:

$$
(((C \cdot x) \cdot y) \cdot z) \cdot u \quad \rightarrow \quad (x \cdot z) \cdot (((x \cdot y) \cdot z) \cdot u) \,.
$$

□

Example 31. The lexicographic path ordering cannot directly handle the following system:

$$
\begin{aligned}
(x \cdot y) \cdot z &\rightarrow x \cdot (y \cdot z) \\
(x + y) \cdot z &\rightarrow (x \cdot z) + (y \cdot z) \\
z \cdot (x + f(y)) &\rightarrow g(z, y) \cdot (x + a) \,.
\end{aligned}
$$

But termination can be proved using a *semantic path ordering* [Kamin and Lévy, 1980] with any term of the form $z \cdot (x + f(y))$ greater than any other product, any product greater than any other term, and term greater than its subterms, and products treated lexicographically (left-to-right). Note that no rule application changes the value of the redex. □

Example 32 (Insertion Sort).

$$
\begin{aligned}
sort(nil) &\rightarrow nil \\
sort(cons(x, y)) &\rightarrow insert(x, sort(y)) \\
insert(x, nil) &\rightarrow cons(x, nil) \\
insert(x, cons(v, w)) &\rightarrow choose(x, cons(v, w), x, v) \\
choose(x, cons(v, w), y, 0) &\rightarrow cons(x, cons(v, w)) \\
choose(x, cons(v, w), 0, s(z)) &\rightarrow cons(v, insert(x, w)) \\
choose(x, cons(v, w), s(y), s(z)) &\rightarrow choose(x, cons(v, w), y, z) \,.
\end{aligned}
$$

Termination can be shown using subterm and the quadruple $\langle \mathbf{top}, \mathbf{argument}, \mathbf{top}, \mathbf{argument}\rangle$ to show that the left side is greater than the right and its subterms. The first precedence is $sort > insert \approx choose > cons$; the second, $insert > choose$. For the second component, we take the first argument of $sort$, second argument of $choose$ and $insert$, and any constant for the others; for the last component, we take the third argument of $choose$. For details of this *general path ordering*, see [Dershowitz and Hoot, to appear]. □

Example 33 (Battle of Hydra and Hercules). The following system, is terminating, but not provably so in Peano Arithmetic:

$$
\begin{aligned}
h(z, e(x)) &\rightarrow h(c(z), d(z, x)) \\
d(z, g(0, 0)) &\rightarrow e(0) \\
d(z, g(x, y)) &\rightarrow g(e(x), d(z, y)) \\
d(c(z), g(g(x, y), 0)) &\rightarrow g(d(c(z), g(x, y)), d(z, g(x, y))) \\
g(e(x), e(y)) &\rightarrow e(g(x, y)) \,.
\end{aligned}
$$

Use a general path ordering, with semantic component

$$
\begin{array}{ll}
[\![g(x, y)]\!] = \omega^{[\![x]\!]} + [\![y]\!] & [\![h(z, x)]\!] = [\![z]\!] + [\![x]\!] \\
[\![d(z, x)]\!] = pred_{[\![z]\!]}([\![x]\!]) & [\![e(x)]\!] = [\![x]\!] \\
[\![c(x)]\!] = [\![x]\!] + 1 & [\![0]\!] = = 1 \,,
\end{array}
$$

and compare $\langle [\![\mathbf{term}]\!], \mathbf{top}, \mathbf{second\ argument\ of}\ d\ \mathbf{and}\ g\rangle$, with precedence $d > g > e$. □

References

[Dershowitz and Hoot, to appear] Nachum Dershowitz and Charles Hoot. Natural termination. *Theoretical Computer Science*, to appear.

[Dershowitz and Jouannaud, 1990] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Methods and Semantics, chapter 6, pages 243–320. North-Holland, Amsterdam, 1990.

[Dershowitz and Manna, 1979] Nachum Dershowitz and Zohar Manna. Proving termination with multiset orderings. *Communications of the ACM*, 22(8):465–476, August 1979.

[Dershowitz, 1982] Nachum Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, March 1982.

[Dershowitz, 1987] Nachum Dershowitz. Termination of rewriting. *J. Symbolic Computation*, 3(1&2):69–115, February/April 1987. Corrigendum: *4*, 3 (December 1987), 409–410; reprinted in *Rewriting Techniques and Applications*, J.-P. Jouannaud, ed., pp. 69—115, Academic Press, 1987.

[Evans, 1951] Trevor Evans. On multiplicative systems defined by generators and relations, I. *Proceedings of the Cambridge Philosophical Society*, 47:637–649, 1951.

[Gorn, 1973] Saul Gorn. On the conclusive validation of symbol manipulation processes (how do you know it has to work?). *J. of the Franklin Institute*, 296(6):499–518, December 1973.

[Iturriaga, 1967] R. Iturriaga. Contributions to mechanical mathematics. Ph.D. Thesis, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, 1967.

[Kamin and Lévy, 1980] 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.

[Klop, 1992] Jan Willem Klop. Term rewriting systems. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2, chapter 1, pages 1–117. Oxford University Press, Oxford, 1992.

[Knuth and Bendix, 1970] Donald E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, Oxford, U. K., 1970. Reprinted in *Automation of Reasoning 2*, Springer-Verlag, Berlin, pp. 342–376 (1983).

[Lankford, 1979] 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.

[Lipton and Snyder, 1977] R. Lipton and L. Snyder. On the halting of tree replacement systems. In *Proceedings of the Conference on Theoretical Computer Science*, pages 43–46, Waterloo, Canada, August 1977.

[Manna and Ness, 1970] Zohar Manna and Steven Ness. On the termination of Markov algorithms. In *Proceedings of the Third Hawaii International Conference on System Science*, pages 789–792, Honolulu, HI, January 1970.

[Plaisted, 1978a] David A. Plaisted. Well-founded orderings for proving termination of systems of rewrite rules. Report R-78-932, Department of Computer Science, University of Illinois, Urbana, IL, July 1978.

[Plaisted, 1978b] David A. Plaisted. A recursively defined ordering for proving termination of term rewriting systems. Report R-78-943, Department of Computer Science, University of Illinois, Urbana, IL, September 1978.

[Plaisted, 1993] David A. Plaisted. Term rewriting systems. In D. M. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 4, chapter 2. Oxford University Press, Oxford, 1993. To appear.

[Zantema, 1991] Hans Zantema. Classifying termination of term rewriting. Technical Report RUU-CS-91-42, Utrecht University, The Netherlands, November 1991.