Higher-Order Recursive Path Orderings

Jean-Pierre Jouannaud† and Albert Rubio‡

† LRI, Bat. 490, CNRS/Université de Paris Sud, 91405 Orsay, FRANCE
Email: Jean-Pierre.Jouannaud@lri.fr
‡ Universitat Politècnica de Catalunya, Pau Gargallo 5, 08028 Barcelona, SPAIN
Email: rubio@lsi.upc.es

Rewrite rules are increasingly used in programming languages and logical systems, with two main goals: defining functions by pattern matching; describing rule-based decision procedures. ML, Elf [9] and Isabelle [8] examplify the first use. A future version of Coq [4] will examplify the second use [3]. In Elf and Isabelle, rules operate on terms in β-normal, η-expanded form. In ML and Coq, they operate on arbitrary terms. In the future version of Coq, both kinds will probably coexist.

Our ambition is to develop for the higher-order case the same kind of semi-automated termination proof techniques that are available for the first-order case, of which the most popular one is the recursive path ordering [2].

Our first important contribution is a reduction ordering for typed higher-order terms which conservatively extends Dershowitz's recursive path ordering for first-order terms. In the latter, the precedence rule allows to decrease from the term $s = f(s_1, \ldots, s_n)$ to the term $g(t_1, \ldots, t_n)$, provided that (i) f is bigger than g in the given precedence on function symbols, and (ii) s is bigger than every t_i. For typing reasons, in our ordering the latter condition becomes: (ii) for every t_i, either s or some s_j is bigger than or equal to t_i. Indeed, we can instead allow t_i to be obtained from the subterms of s by computability preserving operations. Here, computability refers to Tait and Girard's strong normalization proof technique which we have used to show that our ordering is well-founded and compatible with β-reductions. To hint at the strength of the ordering, let us mention that the polymorphic version of Gödel's recursor for the natural numbers is easily oriented. And indeed, our ordering applies to arbitrary polymorphic signatures. Several other examples are given which examplify the expressive power of the ordering.

As a second contribution, we then restrict the previous ordering so as to obtain a reduction ordering operating on terms in η-long β-normal form. Only very weak orderings had been obtained so far for that purpose [6, 7, 1, 5].

References

1. J. Van de Pol and H. Schwichtenberg. Strict functional for termination proofs. In *Typed Lambda Calculi and Applications, Edinburgh*. Springer-Verlag, 1995.
2. Nachum Dershowitz. Orderings for term rewriting systems. *Theoretical Computer Science*, 17(3):279–301, March 1982.

3. G. Dowek, T. Hardin, and C. Kirchner. Theorem proving modulo, 1998. Rapport de recherche INRIA 3400.
4. Gilles Dowek, Amy Felty, Hugo Herbelin, Gérard Huet, Christine Paulin-Mohring, and Benjamin Werner. The coq proof assistant user's guide version 5.6. INRIA Rocquencourt and ENS Lyon.
5. Jean-Pierre Jouannaud and Albert Rubio. Rewrite orderings for higher-order terms in η-long β-normal form and the recursive path ordering. *Theoretical Computer Science*, 208(1–2):3–31, November 1998.
6. C. Loría-Sáenz and J. Steinbach. Termination of combined (rewrite and λ-calculus) systems. In *Proc. 3rd Int. Workshop on Conditional Term Rewriting Systems, Pont-à-Mousson, LNCS 656*, volume 656 of *Lecture Notes in Computer Science*, pages 143–147. Springer-Verlag, 1992.
7. Olav Lysne and Javier Piris. A termination ordering for higher order rewrite systems. In *Proc. 6th Rewriting Techniques and Applications, Kaiserslautern, LNCS 914*, Kaiserslautern, Germany, 1995.
8. Lawrence C. Paulson. Isabelle: the next 700 theorem provers. In P. Odifreddi, editor, *Logic and Computer Science*. Academic Press, 1990.
9. Frank Pfenning. Elf: A language for logic definition and verified meta-programming. In *Proc. 4th IEEE Symp. Logic in Computer Science*, pages 313–322, 1989.

This article was processed using the LaTeX macro package with LLNCS style