# Generating Polynomial Orderings for Termination Proofs*

Jürgen Giesl
FB Informatik, Technische Hochschule Darmstadt,
Alexanderstr. 10, 64283 Darmstadt, Germany
Email: giesl@inferenzsysteme.informatik.th-darmstadt.de

**Abstract.** Most systems for the automation of termination proofs using polynomial orderings are only *semi-automatic*, i.e. the "right" polynomial ordering has to be given by the user. We show that a variation of Lankford's partial derivative technique leads to an easier and slightly more powerful method than most other semi-automatic approaches. Based on this technique we develop a method for the *automated synthesis* of a suited polynomial ordering.

## 1 Introduction

A term rewriting system (trs) $\mathcal{R}$ is *terminating* for a set of terms $\mathcal{T}$ if there exists no infinite derivation of terms in $\mathcal{T}$. While in general this problem is undecidable [HL78], several methods for proving termination have been developed, cf. [Der87]. We present a method for *automated* termination proofs using polynomial orderings, which is based on a variant of Lankford's partial derivative technique (section 2). Our method can be used both in a *semi-automatic* and a *fully automated* way (section 3).

## 2 A Termination Criterion with Variable Coefficients

The use of *polynomial orderings* for termination proofs has been suggested by D. S. Lankford [Lan79] and has been extended to *real* polynomials by N. Dershowitz [Der82]. A *polynomial interpretation* $\tau$ associates a real multivariate polynomial $f_\tau(x_1, \ldots, x_n)$ with each $n$-ary function symbol $f$. The ordering implicitly defined by a polynomial interpretation $\tau$ is called the *corresponding* polynomial ordering $\succ_\tau$ (i.e. $t \succ_\tau s$ iff $\tau(t) > \tau(s)$). To use $\succ_\tau$ for termination proofs, $\succ_\tau$ must be the strict part of a *quasi-simplification ordering* (i.e. $\succeq_\tau$ must be *monotonic* and must satisfy the *subterm property*).

In order to compare *non-ground* terms, $\tau$ is extended to interpret variables as variables over the reals. To prove the termination of a trs $\mathcal{R}$ (with *finitely* many rules), $\mathcal{R}$ has to be *compatible with* a polynomial ordering; i.e. for each rule $l \to r$ in $\mathcal{R}$, $\tau(l) > \tau(r)$ must hold for all instantiations of the variables with numbers $n$ that are *greater or equal than the minimal value of a ground term* (i.e. numbers $n$ with $n \geq \min\{\tau(t) \mid t \text{ ground term}\}$) [DJ90][1].

* This paper appeared in the *Proceedings of the 6th International Conference on Rewriting Techniques and Applications*, Kaiserslautern, Germany, Springer–Verlag, LNCS 914, 1995.

[1] We always assume that there exist ground terms in $\mathcal{T}$.

Consider the trs $\mathcal{R}$ for associativity and endomorphism from [Bel84] and [BL87]. Here $\mathcal{T}$ consists of all terms constructed from the constant a, the unary function symbol map and the binary function symbol $\circ$.

$$(x \circ y) \circ z \to x \circ (y \circ z), \tag{1}$$

$$\mathsf{map}(x) \circ \mathsf{map}(y) \to \mathsf{map}(x \circ y), \tag{2}$$

$$\mathsf{map}(x) \circ (\mathsf{map}(y) \circ z) \to \mathsf{map}(x \circ y) \circ z. \tag{3}$$

To generate a polynomial interpretation we first have to decide on the *maximum degree* of the polynomials. We follow a heuristic from [Ste91] and associate a *simple-mixed*[2] polynomial with each function symbol. So in our example the constant a is associated with a polynomial $a_0$, the unary function symbol map is associated with a polynomial $\mathsf{map}_\tau(x) = m_0 + m_1 x$ (or $m_0 + m_2 x^2$) and $\circ$ is associated with $\circ_\tau(x, y) = c_0 + c_1 x + c_2 y + c_3 xy$. Here we use a polynomial interpretation $\tau$ which maps function symbols to polynomials with *variable coefficients* $a_0, m_0, m_1, c_0, c_1, c_2, c_3$.

Now we have to find an instantiation of the variable coefficients $a_0, \ldots, c_3$ such that $\tau(l) - \tau(r) > 0$ holds for each rule $l \to r$ in $\mathcal{R}$. For the first rule (1) we obtain the following inequality.

$$c_0c_1 - c_0c_2 + (c_1^2 - c_1 - c_0c_3)\,x + (c_2 - c_2^2 + c_0c_3)\,z + (c_1c_3 - c_2c_3)\,xz \;>\; 0. \tag{4}$$

The problem is that we cannot directly check whether an instantiation of the variable coefficients $c_0, \ldots, c_3$ makes this inequality valid *for all* $x, z \geq \min\{\tau(t) \mid t \text{ ground term}\}$. Note that in general this question is undecidable [Lan79]. Therefore we will transform (4) into new inequalities which *do not contain the rule variables* $x$ and $z$ any more. Then for each instantiation of the variable coefficients it is trivial to check whether they satisfy these new inequalities. The invariant of this transformation is that every instantiation of $c_0, \ldots, c_3$ satisfying the new inequalities also satisfies the original inequality *for all* $x, z \geq \min\{\tau(t) \mid t \text{ ground term}\}$.

Let $\mu$ be a new variable and let us assume for the moment that $\mu$ is instantiated with a value less or equal than $\min\{\tau(t) \mid t \text{ ground term}\}$. Then instead of demanding that inequality (4) should hold for all $x, z \geq \mu$, it is sufficient if this inequality holds for $x = \mu$ and if the polynomial on the right hand side of inequality (4) is not decreasing when $x$ is increasing. In other words, the partial derivative of this polynomial with respect to $x$ should be non-negative. Therefore we can replace (4) by the inequalities

$$c_0c_1 - c_0c_2 + (\ldots)\mu + (\ldots)z + (\ldots)\mu z > 0 \;\text{ (resulting from } x = \mu\text{) and} \tag{5}$$

$$c_1^2 - c_1 - c_0c_3 + (c_1c_3 - c_2c_3)z \geq 0 \;\text{ (resulting from partial derivation).} \tag{6}$$

By further application of this technique (i.e. demanding that (5) and (6) hold for $z = \mu$ and that their partial derivatives with respect to $z$ are non-negative) we

[2] A non-unary polynomial $p$ is *simple-mixed* iff all its exponents are not greater than 1. A unary polynomial $p$ is simple-mixed if it has the form $\alpha_0 + \alpha_1 x$ or $\alpha_0 + \alpha_2 x^2$.

obtain inequalities without the variables $x$ and $z$. We proceed analogously for the other rules (2), (3) of $\mathcal{R}$ and obtain inequalities which only contain the variable coefficients $a_0, \ldots, c_3$ and $\mu$, but not the rule variables $x, y, z$. If an instantiation satisfies these inequalities, then the trs $\mathcal{R}$ is compatible with the corresponding polynomial ordering.

For the elimination of the rule variables $x, y, z$ we have repeatedly used the following two *differentiation rules.*

$$\frac{p(\ldots x \ldots) > 0}{p(\ldots \mu \ldots) > 0,\ \frac{\partial p(\ldots x \ldots)}{\partial x} \geq 0} \quad \text{(Diff1)}$$

$$\frac{p(\ldots x \ldots) \geq 0}{p(\ldots \mu \ldots) \geq 0, \frac{\partial p(\ldots x \ldots)}{\partial x} \geq 0} \quad \text{(Diff2)}$$

The differentiation rules (Diff1) and (Diff2) are based on the partial derivative method of Lankford [Lan76]. But Lankford's method can only prove that a polynomial is *eventually positive* (i.e. $p(x_1, \ldots, x_n) > 0$ holds for large enough $x_i$). Note that it is not sufficient for the termination of $\mathcal{R}$ if there exists a polynomial interpretation $\tau$ such that $\tau(l) - \tau(r)$ is eventually positive for each rule $l \to r$ in $\mathcal{R}$. For instance, the trs with the rule $x \to \mathsf{a}$ is not terminating although $\tau(x) - \tau(\mathsf{a})$ is eventually positive for every polynomial interpretation $\tau$.

For $\mathcal{R}$'s termination proof we furthermore have to ensure the *subterm property* and *monotonicity* of the corresponding quasi-ordering. To guarantee the subterm property we demand that $\mathsf{map}_\tau(x) - x \geq 0$ (and the corresponding inequalities for $\circ_\tau$) hold and eliminate the variables $x, y$ by application of the differentiation rule (Diff2).

For the monotonicity, we have to ensure that if $x$ is increasing, $\circ_\tau(x, y)$ is not decreasing. So we demand that the partial derivative of $\circ_\tau(x, y)$ with respect to $x$ is non-negative. In our example we have $\frac{\partial \circ_\tau(x,y)}{\partial x} = c_1 + c_3 y$ and therefore we demand $c_1 + c_3 y \geq 0$. Now we can use (Diff2) again to eliminate the remaining rule variable $y$. We proceed analogously for $\mathsf{map}_\tau$ and for the the monotonicity of $\circ_\tau(x, y)$ in its second argument.

Still we have to ensure that the variable $\mu$ is really instantiated with a value less or equal than the minimal value of a ground term. Because of the subterm property, the requirement $\mu \leq \min\{\tau(t) \mid t \text{ ground term}\}$ is equivalent to the condition $\mu \leq c_\tau$ for all constants $c$ of the signature. Therefore in our example the instantiation of the variables also has to satisfy the inequality $a_0 - \mu \geq 0$. The following theorem summarizes our termination criterion using polynomial interpretations with (possibly variable) coefficients.

**Theorem 1 (Termination Criterion with Real Variable Coefficients).** *Let $\mathcal{R}$ be a trs, let $\tau$ be a polynomial interpretation with (possibly variable) coefficients. Repeated application of (Diff1) and (Diff2) to*

$$\begin{aligned}
\tau(l) - \tau(r) &> 0 \quad \textit{for all rules } l \to r \textit{ in } \mathcal{R},\\
f_\tau(\ldots x \ldots) - x &\geq 0 \quad \textit{for all function symbols } f,\\
\tfrac{\partial f_\tau(\ldots x \ldots)}{\partial x} &\geq 0 \quad \textit{for all function symbols } f,\\
c_\tau - \mu &\geq 0 \quad \textit{for all constants } c
\end{aligned}$$

*yields a unique set of inequalities containing no rule variables any more. If there exists an instantiation of the variable coefficients and the variable $\mu$ with real numbers which satisfies the resulting inequalities, then $\mathcal{R}$ is terminating.*

In our example the resulting inequalities are satisfied by the instantiation $\mu = 2$, $a_0 = 2$, $m_0 = 0$, $m_1 = 2$, $c_0 = 0$, $c_1 = 1$, $c_2 = 0$, $c_3 = 1$. (This corresponds to the polynomial interpretation given by $\mathsf{a}_\tau = 2$, $\mathsf{map}_\tau(x) = 2x$ and $\circ_\tau(x, y) = xy + x$.) Therefore by theorem 1 the termination of $\mathcal{R}$ is proved.

Most systems for "automated" termination proofs using polynomial orderings are *semi-automatic*, i.e. the user has to provide a polynomial interpretation and the system checks whether the trs is compatible with the corresponding polynomial ordering. Of course the termination criterion of theorem 1 can also be applied in a semi-automatic way. Then we use associations to polynomials whose coefficients are *numbers* (instead of variables) and we replace the variable $\mu$ by a *number* $\mu$. A comparison with the semi-automatic methods of Ben Cherifa and Lescanne [BL87] and Steinbach [Ste92] leads to the following results[3].

- *If [Ste92] and [BL87] can prove a polynomial $p$ positive, then our method can do so as well.*
- *If our method can prove $p$ positive for all $x_1, \ldots, x_n \geq \mu$, then there exists a $\mu' \geq \mu$ such that the methods of [Ste92] and [BL87] can prove $p$ positive for all $x_1, \ldots, x_n \geq \mu'$. Choosing $\mu' = \mu$ is not always possible.*
- *While the worst case complexity of the systems in [Ste92] and [BL87] is exponential in the number of monomials in $p$, our method is exponential in the number of its variables.*

## 3 A Fully Automated Termination Proof Procedure

In theorem 1 we introduced a method to automatically generate a set of inequalities only containing variable coefficients and the variable $\mu$. To prove the termination of a trs $\mathcal{R}$ mechanically we now have to synthesize an instantiation of these variables satisfying the inequalities.

When examining term rewriting systems occurring in the literature we noticed that most termination proofs with polynomial interpretations only use polynomials whose coefficients are 0, 1 or 2. Checking whether a certain instantiation of variables with numbers satisfies the inequalities resulting from theorem 1 can be done very efficiently. Therefore we suggest to apply a "generate and test" approach first which generates all instantiations of the variables with numbers from $\{0, 1, 2\}$ until one of these instantiations satisfies the inequalities. This results in a fully automated termination proof procedure which succeeds for most of those term rewriting systems which are compatible with a polynomial ordering.

[3] In this comparison we do not consider the additional use of the arithmetic-mean-geometric-mean inequality in [Ste92] and extend the method of [BL87] by backtracking and arbitrary minimal value $\mu$.

Nevertheless there do exist term rewriting systems which require a polynomial ordering with coefficients other than 0, 1 or 2. It is decidable whether there exists an instantiation with real numbers satisfying a set of inequalities [Tar51]. But even the most efficient known decision method (the *cylindrical algebraic decomposition* algorithm by G. E. Collins [Col75]) is very time-consuming. For that reason these methods have been rarely used for automated termination proofs.

Therefore we suggest an *incomplete*, more efficient modification of Collins' algorithm. As we know of no trs whose termination proof requires a polynomial interpretation with non-rational real coefficients, we have restricted the algorithm to *rational* instead of *real* numbers which eases the implementation considerably. Moreover, we have introduced execution time limits for each step of Collins' algorithm. If the time limit for the actual step is exceeded, then the algorithm can only use the results of the actual step computed so far and has to carry on with the next step. Now Collins' algorithm is no longer used as a decision method, but only as a *heuristic*.

To sum up, we propose the following termination proof procedure:

1. Construct a set of inequalities as described in theorem 1 (using a polynomial interpretation with possibly variable coefficients).
2. Check whether these inequalities are satisfied by an instantiation with numbers from $\{0, 1, 2\}$.
3. If not, try to prove their satisfiability by a modified version of Collins' algorithm.

Instead of the differentiation rules we could also use a technique from [Ste92] for the elimination of the rule variables $x, y, z$. But while Steinbach's technique introduces *several* new variables, the advantage of (Diff1) and (Diff2) is that these rules introduce only *one* new variable $\mu$. For the generation of a polynomial ordering compatible with $\mathcal{R}$ we therefore only have to find an instantiation of the variable coefficients and $\mu$.

An alternative approach for the automated generation of the "right" polynomial interpretation has been presented in [Ste91] which can be useful if the number of variable coefficients is small. In these cases Steinbach's method may also be used to search for an instantiation that satisfies the inequalities resulting from theorem 1.

We have presented an efficient, powerful and easy to implement algorithm for termination proofs using polynomial orderings which can be used both in a semi-automatic and in a fully automated way. Our termination proof procedure has been implemented[4] in Common Lisp on a Sun SPARC-2. Table 1 illustrates its performance with some examples. The second row contains the execution time our algorithm needs to generate a polynomial interpretation which is compatible with the trs in the first row.

[4] The implementation and an extended version of this paper are available by anonymous ftp from `kirmes.inferenzsysteme.informatik.th-darmstadt.de` under `pub/termination`.

| Example | Time |
|---|---|
| Nested Function Symbols ([Ste91, Example 8.1]) | 0.1 sec. |
| Endomorphism & Associativity ([Bel84], [BL87]) | 0.1 sec. |
| Running Example 6.1 in [Ste91] (by A. Middeldorp) | 0.2 sec. |
| Binomial Coefficients ([Ste91, Example 8.8], [Ste92, Example 13]) | 1.6 sec. |
| Distributivity & Associativity ([Der87, p. 78]) | 1.9 sec. |

**Table 1.** Performance of our method.

**Acknowledgements**
I would like to thank Jürgen Brauburger, Caroline Claus, Alexander Friedl, Stefan Gerberding, Thomas Kolbe, Jens Marschner, Martin Protzen and Christoph Walther for comments and support.

## References

[Bel84] F. Bellegarde. Rewriting Systems on FP Expressions that reduce the Number of Sequences they yield. *Symp. LISP & Funct. Prog.*, ACM, Austin, TX, 1984.

[BL87] A. Ben Cherifa & P. Lescanne. Termination of Rewriting Systems by Polynomial Interpretations and its Implementation. *Science of Computer Programming*, 9(2):137-159, 1987.

[Col75] G. E. Collins. Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition. In *Proc. 2nd GI Conf. on Automata Theory and Formal Languages*, Kaiserslautern, Germany, 1975.

[Der82] N. Dershowitz. Orderings for Term-Rewriting Systems. *Theoretical Computer Science*, 17:279-301, 1982.

[Der87] N. Dershowitz. Termination of Rewriting. *Journal of Symbolic Computation*, 3(1, 2):69-115, 1987.

[DJ90] N. Dershowitz & J.-P. Jouannaud. Rewrite Systems. *Handbook of Theoretical Comp. Science*, J. van Leuwen, Ed., vol. B, ch. 6, 243-320, Elsevier, 1990.

[HL78] G. Huet & D. S. Lankford. On the Uniform Halting Problem for Term Rewriting Systems. Rapport Laboria 283, Institut de Recherche d'Informatique et d'Automatique, Le Chesnay, France, 1978.

[Lan76] D. S. Lankford. A Finite Termination Algorithm. Internal Memo, Southwestern University, Georgetown, TX, 1976.

[Lan79] D. S. Lankford. On Proving Term Rewriting Systems are Noetherian. Technical Report Memo MTP-3, Louisiana Tech. Univ., Ruston, LA, 1979.

[Ste91] J. Steinbach. Termination Proofs of Rewriting Systems — Heuristics for Generating Polynomial Orderings. SEKI-Report SR-91-14, Univ. Kaiserslautern, Germany, 1991.

[Ste92] J. Steinbach. Proving Polynomials Positive. In *Proc. 12th Conf. Foundations Software Technology & Theoretical Comp. Sc.*, New Delhi, India, 1992.

[Tar51] A. Tarski. *A Decision Method for Elementary Algebra and Geometry.* University of California Press, Berkeley, 1951.