プログラミングー言語・基礎・実践ー 19ー2
（1994. 11. 4）

ギャップ条件を持つ単純停止性

小川 瑞史
mizuhito@ntt-20.ntt.jp
ＮＴＴ基礎研究所

梗概

本報告では単純停止性の拡張である、ギャップ条件を持つ単純停止性を提案し、構成子を共有する場合のモジュラー性を示す。ギャップ条件を持つ単純停止性は **Kruskal** の定理の **Friedman** による拡張に基づく。さらに multiset path ordering（status を持たない再帰的経路順序）の拡張の試みとして、$\succcurlyeq_{mgo}$ を定義し、$f(f(x)) \to f(g(f(x)))$ の停止性を示す。しかし Puel の SRPO と比べ十分に強力とはいえないので順序のデザインは今後の課題である。

Simple termination with gap-condition

Mizuhito Ogawa
mizuhito@ntt-20.ntt.jp
NTT Basic Research Labos.

Abstract

This paper reports an extension of simple termination, called *simple gap termination*, based on Kruskal's theorem with gap-condition. Its modularity (with shared constructors) is also shown.
A trial to design an ordering $\succcurlyeq_{mgo}$, which is an extension of multiset path ordering, is also given. However, this ordering is neither a strict ordering nor powerful compared with Puel's *SRPO*. Further investigation is needed for designing orderings.

1 Introduction

A Term Rewriting System (TRS, for short) is a set of directed equations, and is widely applied for a computational model, theorem proving, etc. Two important properties of a TRS are *confluence* and *termination*. Frequently used method to show termination is *simple termination* [N.D82, J.W92]. Simple termination has good properties:

1. Simple termination satisfies modularity (with shared constructors) [MA92]. (i.e, For any pair of TRSs R_1 and R_2 s.t. their all common function symbols are constructors, R_1 and R_2 are simply terminating if and only if $R_1 \cup R_2$ is simply terminating.)

2. Simple termination includes practically useful precedence-based term orderings. Namely, LPO (Lexicographic Path Ordering), RPO (Recursive Path Ordering), etc. [N.D87, M.R87] These orderings have automatic procedures for proving termination [DR85][1].

Simple termination is a practically powerful method, but fails in following cases.

- Simple termination is based on Kruskal's theorem [J.K60, CW63], thus simple termination cannot treat a rule in which the lhs is embedded into the rhs, such as $f(f(x)) \to f(g(f(x)))$.

- Precedence-based term orderings may cause conflicts on precedence. For instance, an addition $+$ in Cohen-Watson system for integer arithmetic [DP91] cannot be proved by precedence-based term orderings.

- Frequently used RPO (with status) may cause conflicts on status. For instance, *explicit substitution* in TRS format [H.Z94] cannot be proved by RPO.

[1]Though termination and simple termination are undecidable even for an one-rule TRS [AC91, AB93].

For the first tree-embedding case, *S-embedding* based on Higman's lemma with unavoidable patterns [L.P89, P.L92] is quite effective. An S-embedding is based on a precedence on unavoidable patterns instead of a precedence on function symbols. A set of unavoidable patterns is a set of patterns which will match to almost every terms (i.e., except for finitely many terms). For instance, patterns $\{f(\Box), g(f(\Box)), g(g(\Box))\}$ with a precedence $g(f(\Box)) < f(\Box)$ lead the termination of $f(f(x)) \to f(g(f(x)))$ by RPO-like manner. Unfortunately, modularity is unclear in her method.

For all cases, *semantic labeling* [H.Z94] is useful. Semantic labeling distinguishes occurrences of a function symbol under a suitable model - then a function symbol is labeled with a value of its subterms. This avoids conflicts in precedence and obtains freedom to determine precedence. Semantic labeling is too strong in some sense - A TRS is terminating if and only if there exists a suitable semantic labeling. Thus, semantic labeling lost both modularity[Y.T87] and an automatic termination detection.

This paper proposes *simple gap termination*, which is a proper extension of simple termination. This extension is based on an extension of Kruskal's theorem due to H.Friedman, called *Kruskal's theorem with gap-condition* [S.G85, I.K89, L.G90] (other than Higman's lemma with unavoidable patterns in [L.P89]). Kruskal's theorem with gap-condition employs a tree-embedding $\psi : s \to t$ which satisfies $\psi(succ(a)) \leq b$ for each vertex a in s and each vertex b of t s.t. $\psi(a) < b \leq \psi(succ(a))$, under some total precedence $<$ on function symbols. The basic idea is to replace a condition $f(\cdots, s, \cdots) \succ s$ in simple termination with $C[\cdots, s, \cdots] \succ s$ if each function symbol f on a path from the root of $C[\cdots, s, \cdots]$ to the root of s satisfies $f \geq root(s)$. The modularity (with shared constructors) of simple gap termination is shown similarly to simple termination [MA92].

A trial to design an extension of term orderings gives *MGO (Multiset Gap Ordering)* $\succ_{mgo}$, and this can prove the termination of $f(f(x)) \to f(g(f(x)))$. However, this ordering is neither a strict ordering nor powerful compared with Puel's *SRPO*. Further investigation is needed for designing orderings.

2 Preliminaries

Let F be a set of *function symbols* and V a countably infinite set of *variables* s.t. F and V are disjoint each other. For every $f \in F$, a natural number *arity* is associated with f. Function symbols with arity 0 is called *constant*. The set of all *terms* built from F and V is defined as usual. The set of variables occurring in a term t is denoted by $V(t)$. A term t is said to be *ground* if $V(t) = \emptyset$. A term t without repeated occurrence of a variable is said to be *linear*.

A *substitution* is a map from variables to terms and the domain is naturally expanded to whole terms. Application of a substitution σ to a term t is written as $t\sigma$. A substitution σ is also written as $\{x_1 := t_1, \ldots, x_n := t_n\}$, where x_i's are variables s.t. $x_i\sigma \not\equiv x_i$.

Let $\Box$ be a special constant symbol. A context $C[\,]$ is a term in $T(F \cup \Box, V)$. When $C[\,]$ is a context with n $\Box$'s and $t_1, \cdots, t_n$ are terms, $C[t_1, \cdots, t_n]$ denotes the term obtained by replacing all $\Box$'s in $C[\,]$ with t_i's in left-to-right manner. A term t is called a *subterm* of a term s if there is a context $C[\,]$ s.t. $C[t] \equiv s$.

The set of positions $P(t)$ of a term t is defined as below:

1. $P(t) = \Lambda$ if t is either a constant or a variable.

2. $P(t) = \{\Lambda\} \cup \{i \cdot u \mid 1 \leq i \leq n \text{ and } u \in P(t_i)\}$ if $t \equiv f(t_1, \ldots, t_n)$.

For a position $p \in P(t)$, t/p is the subterm occurring at p. The set $\{p \mid t/p \in V\}$ is denoted by $P_V(t)$. For terms t, s and a position $p \in P(t)$, $t[p \leftarrow s]$ is the term obtained by replacing the subterm at p in t with s.

For positions p_1, p_2, we write $p_1 \leq p_2$ if p_1 is a prefix of p_2, and $p_1 \perp p_2$ if neither $p_1 \leq p_2$ nor $p_2 \leq p_1$. The longest common prefix of p_1 and p_2 is denoted by $\wedge(p_1, p_2)$. The concatenation of sequences p_1 and p_2 is $p_1 \cdot p_2$. If $p_1 < p_2$, $p_1 \backslash p_2$ is a sequence which is obtained from p_2 by removing its prefix p_1.

A *reduction system* $\rightarrow$ is a binary relation, i.e. a set of pairs of an underlying domain. An element of a reduction system is called a *reduction*, and denoted by $a \rightarrow a'$. A symmetric closure, reflexive transitive closure, and reflexive transitive symmetric closure of $\rightarrow$ are written as $\leftrightarrow$, $\rightarrow^*$ and, $\leftrightarrow^*$, respectively. We also call $\leftrightarrow$ a reduction. If there is no reduction s.t. $a \rightarrow a'$, a is a *normal form* of a reduction system.

Definition 2.1 A reduction system $\rightarrow$ is *terminating* if there is no infinite sequence s.t. $a_1 \rightarrow a_2 \rightarrow \cdots$.

A *term rewriting system* (TRS, for short) R is a finite set of rewrite rules. A *rewrite rule* $l \rightarrow r$ is a pair of terms l, r satisfying following properties:

1. l is not a variable,

2. $V(l) \supseteq V(r)$.

A reduction system $\rightarrow_R$ on the set of terms is defined from a TRS R as:

$$\rightarrow_R = \{C[l\theta] \rightarrow_R C[r\theta] \mid C[\,] \text{ is a context}, \theta \text{ is a substitution, and } l \rightarrow r \in R\}$$

For $\alpha : s \rightarrow t$, a position where a reduction rule is applied is denoted by $p(\alpha)$. We call $l\theta$ a *redex* of R. We often do not distinguish a term rewriting system R and a reduction system $\rightarrow_R$.

3 Simple gap-termination

3.1 Kruskal's theorem with gap-condition

Definition 3.1 A transitive binary relation R on an objective set A is called an *order*.

- If an order R is reflexive, R is called a *quasi-order* (QO, for short).

- If an order R is irreflexive, R is said to be *strict*.

- If an order R is antisymmetric, R is called a *partial order.*
- If each pair of different elements in A is comparable by an order R, R is said to be *total.*
- If a partial order R is total, R is called a *linear order.*

The notations are as in Table 1.

Lemma 3.1 If an order $\succcurlyeq$ is strict, $\succcurlyeq$ is a partial order.

Proof Assume $\succcurlyeq$ is not antisymmetric. Then, there exists s, y $(x \neq y)$ s.t. $x \succcurlyeq y$ and $x \preccurlyeq y$. Then, from transitivity, $x \succcurlyeq y \succcurlyeq x$ implies $x \succcurlyeq x$. This is contradiction. ∎

Lemma 3.2 If $\succcurlyeq$ is an order, $\succ$ is a strict order.

Proof We will show that $s \succcurlyeq t \succ u$ implies $s \succ u$. If $s \succcurlyeq t \succ u$, $s \succcurlyeq t \succcurlyeq u$. Thus, $s \succcurlyeq u$. Assume $s \preccurlyeq u$. Then, since $s \succcurlyeq t$, $t \preccurlyeq u$. This contradicts to $t \succ u$, and $\succ$ is an order. Furthermore $\succ$ is obviously irreflexive, thus strict. ∎

Definition 3.2 An infinite sequence $a_1, a_2, \cdots$ of A is *good* if there exist i, j s.t. $i < j$ and $a_i \leq a_j$. An infinite sequence $a_1, a_2, \cdots$ is *bad* if $a_1, a_2, \cdots$ is not good. A QO $(A, \sqsubseteq)$ is a *well quasi-order* (WQO, for short) if every infinite sequence of A is good.

Definition 3.3 A pair of vertices v, v' of a tree t satisfy $v \leq v'$ if v is in a path from the root of t and v'. We denote $v < v'$ if $v \leq v'$ and $v \neq v'$, and $v = parent(v')$ if v is the maximum vertex s.t. $v < v'$. We call v an *infima* of v_1, v_2 if v is the maximum vertex s.t. $v < v_1, v_2$. A *tree embedding* $\phi: s \rightarrow t$ is a strictly increasing (one-to-one) mapping from vertices of s to those that of t preserving each existing infima.

Theorem 3.1 Let $\leq$ be a WQO on a set F of labels, and let $T(F)$ be the set of all finite trees with labels from F. Then, $\leq_t$ is a WQO on the set $T(F)$, where $s \leq_t t$ if there exists a tree embedding $\phi: s \rightarrow t$ s.t. $v \leq \phi(v)$ for each vertex v of s.

Several extensions, called Kruskal's theorem with gap-condition, have been proposed in literatures [S.G85, I.K89, L.G90]. We employ the original form by H.Friedman [S.G85].

Theorem 3.2 For $n < \omega$, $T(n)$ is the set of all finite trees with labels less-than-equal n. Then $\leq_g$ is a WQO on the set $T(n)$, where $s \leq_g t$ if there exists a tree embedding $\phi: s \rightarrow t$ s.t.

1. $label(v) = label(\phi(v))$ for each vertex v of s.
2. Let v, v' be vertices s.t. v' is an immediate successor of v. Then, $label(w) \geq label(v')$ for each w s.t. $\psi(v) < w < \psi(v')$.

Corollary 3.1 Notations are same as in the theorem. $\leq_G$ is a WQO on the set $T(n)$ where $s \leq_G t$ if there exists a tree embedding $\phi: s \rightarrow t$ s.t.

1. If v is a leaf vertex of s, $\phi(v)$ is a leaf vertex of t.
2. $label(v) = label(\phi(v))$ for each vertex v of s.
3. Let v, v' be vertices s.t. v' is an immediate successor of v. Then, $label(w) \geq label(v')$ for each w s.t. $\psi(v) < w < \psi(v')$.
4. $label(w) \geq label(root(s))$ for each w s.t. $w < \psi(root(s))$.

Proof Let t^+ be a tree obtained from t by adding 1 to each label. Let $\bar{t}$ be a tree obtained from t^+ by (1) adding the new root vertex labeled 0 and its only child vertex is the original vertex, and (2) adding a new leaf vertex labeled $n+2$ to each original leaf vertex.

Let $t_1, t_2, \cdots$ be an infinite sequence of trees in $T(n)$. Then, $\bar{t_1}, \bar{t_2}, \cdots$ is an infinite sequence in $T(n+2)$. Thus from theorem 3.2, there exists a pair $\bar{t_i}$ and $\bar{t_j}$ s.t. $i < j$ and $\bar{t_i} \leq_g \bar{t_j}$. This implies $t_i \leq_G t_j$. ∎

There are two variants of its extensions [I.K89, L.G90] for labels of infinite ordinals.

	R	R^{-1}	$R\setminus R^{-1}$	$R^{-1}\setminus R$	$R\cup =$	$R^{-1}\cup =$	$R\cap R^{-1}$	$(R\cap R^{-1})\cup =$
order	$\succcurlyeq$	$\preccurlyeq$	$\succ$	$\prec$	$\succeq$	$\preceq$	$\approx$	$\approxeq$
quasi-order	$\sqsupseteq$	$\sqsubseteq$	$\sqsupset$	$\sqsubset$	$\sqsupseteq$	$\sqsubseteq$	$\equiv$	$\equiv$
partial order	$\geq$	$\leq$	$>$	$<$	$\geq$	$\leq$	$=$	$=$

Table 1: Notations for orderings

3.2 Simple gap-termination

Definition 3.4 An ordering $\succcurlyeq$ is *monotonic* if for each context $C[\ ]$ $s \succcurlyeq t$ implies $C[s] \succcurlyeq C[t]$ for each context $C[\ ]$. An ordering $\succcurlyeq$ is *stable* if for each substitution θ $s \succcurlyeq t$ implies $s\theta \succcurlyeq t\theta$.

Definition 3.5 The relations $\rightarrow_{sub}$ and $\rightarrow_{gap}$ on terms are defined below:

- $s \rightarrow_{sub} t$ iff $s = C[t]$.
- $s \rightarrow_{gap} t$ iff $s = C[t]$ and $f \geq root(t)$ for all f on a path from the root of $C[t]$ to the root of the proper subterm t.

Definition 3.6 [N.D82] A monotonic strict order $\succ$ is a *simplification ordering* for a set of ground terms T if $s \rightarrow_{sub} t$ possesses $s \succ t$.

Theorem 3.3 [N.D82] Let R be a TRS. If there exists a simplification ordering $\succ$ over the set of terms T s.t.

$$l\theta \succ r\theta$$

for each rule $l \rightarrow r \in R$ and each ground substitution θ, then R is terminating.

Definition 3.7 Let F be a finite set of function symbols and let $>$ be a linear order over F. A monotonic strict order $\succ$ over a set of ground terms $T(F)$ is a *simplification gap-ordering* if $s \rightarrow_{gap} t$ possesses $s \succ t$.

Theorem 3.4 Let F be a finite set of function symbols and let $>$ be a linear order over F. Let R be a TRS. If there exists a simplification gap ordering $\succ$ over the set of terms T s.t.

$$l\theta \succ r\theta$$

for each rule $l \rightarrow r \in R$ and each ground substitution θ, then R is terminating.

Proof Assume there exists an infinite reduction sequence $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \cdots$ of terms s_i. Since variables in the rhs of each reduction rule are included in its lhs, a set V_s of variables appears in $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \cdots$ is finite. Let θ be an arbitrary ground substitution for V_s. Then, $s_1\theta \rightarrow s_2\theta \rightarrow s_3\theta \rightarrow \cdots$ is an infinite reduction sequence of ground terms. Since $\succ$ satisfies $l\theta \succ r\theta$, there exists an infinite descending chain $s_1\theta \succ s_2\theta \succ s_3\theta \succ \cdots$. However, there exist i, j s.t. $i < j$ and $s_i \leq_G s_j$ from corollary 3.1, and $s \leq_G t$ for terms (i.e., with a fixed arity) is inductively obtained by

- There exists u s.t. $s \rightarrow_{gap} u$ and $u \leq_g t$.
- If $s = f(s_1, \cdots, s_n)$ and $t = f(t_1, \cdots, t_n)$, $s_i \leq_g t_i$ for all i.

Since $\succ$ is the simplification gap-ordering, $s_i \leq_g s_j$ implies $s_i \prec s_j$ or $s_i = s_j$. This contradicts to $s_i \succ s_j$. ∎

Theorem 3.5 Simple gap-termination is modular (with shared constructors).

Proof Let $\rightarrow_{\bar{sub}}$ and $\rightarrow_{\bar{gap}}$ be contextual closures of $\rightarrow_{sub}$ and $\rightarrow_{gap}$. The proof is obtained from the proof of the modularity of simple termination in [MA92], by replacing $\rightarrow_{\bar{sub}}$ with $\rightarrow_{\bar{gap}}$. ∎

4 Designing orderings

4.1 LPO and RPO

Definition 4.1 Let F be a set of function symbols. A partial ordering $>$ over F is called a *precedence*. If $>$ is total, $>$ is called a *total precedence*.

Definition 4.2 [N.D82] Let $<$ be a precedence on a set F of function symbols. A *multiset path ordering* (for short, *MPO*) $\succ_{mpo}$ is inductively defined as follows: For a pair of terms $s = f(s_1, \cdots, s_m)$, $t = g(t_1, \cdots, t_n)$, $s \succ_{mpo} t$ if either of cases below.

1. If $f > g$, $s \succ_{mpo} t_j$ for all j.
2. If $f = g$, $[s_1, \cdots, s_m] \succ\!\!\succ_{mpo} [t_1, \cdots, t_n]$ where $\succ\!\!\succ_{mpo}$ is a multiset extension of $\succ_{mpo}$.
3. There exists i s.t. $s_i \succeq_{mpo} t$.

Definition 4.3 Let $<$ be a precedence on a set F of function symbols, and $status(f) \in \{left, right, multi\}$ for each $f \in F$. A *recursive path ordering* (for short, *RPO*) $\succ_{rpo}$ is inductively defined as follows: For a pair of terms $s = f(s_1, \cdots, s_m)$, $t = g(t_1, \cdots, t_n)$, $s \succ_{rpo} t$ if either of cases below.

1. If $f > g$, $s \succ_{rpo} t_j$ for all j.
2. If $f = g$,
   - $status(f) = multi$ and $[s_1, \cdots, s_m] \succ\!\!\succ_{rpo} [t_1, \cdots, t_n]$ where $\succ\!\!\succ_{rpo}$ is a multiset extension of $\succ_{rpo}$.
   - $status(f) = left$, $s \succeq_{rpo} t_j$ for all j, and there exists i s.t. $s_1 = t_1, \cdots, s_{i-1} = t_{i-1}$ and $s_i \succ_{rpo} t_i$.
   - $status(f) = right$, $s \succeq_{rpo} t_j$ for all j, and there exists i s.t. $s_n = t_n, \cdots, s_{i+1} = t_{i+1}$ and $s_i \succ_{rpo} t_i$.
3. There exists i s.t. $s_i \succ_{RPO} t$.

Lemma 4.1 [J.S89] For a total precedence $>$, RPO are linear orders (up to permutations).

Lemma 4.2 [J.S89] Let $>$ be a partial precedence. If $s \succ_{rpo} t$ then there exists a total precedence $>'$ s.t. $>' \supseteq >$ and $s \succ'_{rpo} t$.

Thus, in practice a total precedence is enough. Under a total precedence, most of known simplification orderings - PSO (Path of Subterms Ordering), PDO (Recursive Decomposition Ordering), PSD (Path of Subterms ordering on Decompositions), etc. - are unified. Actually, PSO is equivalent to RPO under a total precedence [M.R87], and PDO, PSD are equivalent to RPO on ground terms under a total precedence [J.S89]. Since existential fragments of RPO are decidable [JPM91], RPO can be extended to RPO* by $s >_{RPO^*} t$ iff $\theta.s\theta >_{RPO} t\theta$ for all ground substitutions θ. Thus, they can be unified for general terms.

4.2 Designing orders

In this section, we will show a trial to design simplification gap-orderings.

Theorem 4.1 [N.D87] A total monotonic ordering $\succ$ is well-founded for derivations, if and only if it is simplification ordering.

This is because $t \succ C[t]$ leads $t \succ C[t] \succ C[C[t]] \succ \cdots$ from monotonicity and strictness. This contradicts to well-foundedness. Thus, weaker restrictions than a simplification ordering must lose either monotonicity, strictness, or totality. Actually, Puel's *SRPO* [L.P89] - which is quite powerful - loses both monotonicity and stability, and requires complex arguments. The next example *MGO* is a quasi-order defined from a total precedence.

Definition 4.4 Let $<$ be a total precedence on a set F of function symbols. Let ϵ be a fresh unary function symbol s.t. ϵ is the least element wrt $<$, and let F_1 and F_2 be a partition of F s.t. $\epsilon \in F_1$. (i.e., F is a disjoint union of $F_1$and F_2.) A *multiset gap ordering* (for short, *MGO*) $\succcurlyeq_{mgo}$ is defined for a pair of terms $s = f(s_1, \cdots, s_m)$, $t = g(t_1, \cdots, t_n)$ s.t. $s \succcurlyeq_{mgo} t$ if either of cases below.

1. If $f > g$, $s \succ_{mgo} t_j$ for all j.
2. If $f = g$, $[s_1, \cdots, s_m] \succ\!\!\succ_{rpo} [t_1, \cdots, t_n]$ where $\succ\!\!\succ_{mgo}$ is a multiset extension of $\succ_{mgo}$.
3. If $f \geq g$ or $f \in F_2$, there exists i s.t. $s_i \succeq_{mgo} t$ for all j.
4. If $f > g$ and $g \in F_1$, $s \succeq_{mgo} t_j$ for all j.

Lemma 4.3 Let $>$ be a total precedence over F.

1. $\succcurlyeq_{mgo}$ is transitive.

2. $\succcurlyeq_{mgo}$ are monotonic.

3. $\succcurlyeq_{mgo}$ is stable.

Lemma 4.4 Let $F^+ = \{\bot, \top\} \cup F$ and a precedence $>$ on F^+ is an extension of a total precedence $>$ on F with $\bot$ as the least unary function symbol and $\top$ as the maximal constant symbol. Let $\succ_{mgo}=\succcurlyeq_{mgo} - \preccurlyeq_{mgo}$ and let $\theta_\top$ be a substitution s.t. $x\theta_\top = \top$ for each variable x. Then,

$$\left\{ \begin{array}{ccc} s & \succ_{mgo} & t. \\ \bot(s) & \succ_{mgo} & \bot(t). \\ s\theta_\top & \succ_{mgo} & t\theta_\top. \end{array} \right.$$

imply $C[s\sigma] \succ_{mgo} C[t\sigma]$ for each context $C[\ ]$ and each substitution σ.

Proofs for these lemmas are due to the induction on $\lambda(s)+\lambda(t)+\lambda(u)$ and case analysis, where $\lambda(s)$ is the length of the term s. These lemmas show the termination of $\{f(f(x)) \to f(g(f(x)))\}$ by $\succ_{mgo}$.

Example 4.1 For $f(f(x)) \to f(g(f(x)))$, take the total precedence as $f > g$, $F_1 = \{g\}$ and $F_2 = \{f\}$.

5 Conclusion

This paper reported an extension of simple termination, called *simple gap termination*, based on Kruskal's theorem with gap-condition. Its modularity (with shared constructors) was also shown.

A trial to design gap-orderings $\succcurlyeq_{mgo}$, which is an extension of $\succ_{mpo}$, was also shown. However, this ordering is neither a strict ordering nor powerful compared with Puel's *SRPO*. Further investigation is needed for designing ordering.

References

[AB93] A.Middeldorp and B.Gramlich. Simple termination is difficult. In *Proc. RTA '93*, pp. 228–242, 1993. LNCS 690.

[AC91] A.-C.Caron. Linear bounded automata and rewrite rules: Influence of initial configurations on decision properties. In *Proc. Int. Conf. on Theory and Practice of Software Development*, pp. 74–89, 1991. LNCS 493.

[CW63] C.ST.J.A.Nash-Williams. On well-quasi-ordering on finite trees. *Proc. Cambridge Phil. Soc.*, Vol. 59, pp. 833–835, 1963.

[DP91] D.Cohen and P.Watson. An efficient representation of arithmetic for term rewriting. In *Proc. RTA '91*, pp. 240–251, 1991. LNCS 488.

[DR85] D.Detlefs and R.Forgaard. A procedure for automatically proving the termination of a set of rewrite rules. In *Proc. RTA '85*, pp. 255–270, 1985. LNCS 202.

[H.Z94] H.Zantema. Termination of term rewriting by semantic labelling, 1994. *preprint*.

[I.K89] I.Kříž. Well-quasiordering finite trees with gap-condition. proof of harvey friedman's conjecture. *Annals of Mathematics*, Vol. 130, pp. 215–226, 1989.

[J.K60] J.Kruskal. Well-quasiordering, the tree theorem, and vazsonyi's conjecture. *Trans. AMS*, Vol. 95, pp. 210–223, 1960.

[JPM91] J.-P.Jouannaud and M.Okada. Satisfiability of systems of ordinal nottions with the subterm property is deciadable. In *Proc. 18th International Colloquium on Automata, Languages and Programming*, pp. 455–468, 1991. LNCS 510.

[J.S89] J.Steinbach. Extensions and comparison of simplification orderings. In *Proc. RTA '89*, pp. 434–448, 1989. LNCS 355.

[J.W92] J.W.Klop. Term rewriting systems. In D.Gabbay S.Abramsky and T.Mainbaum, editors, *Handbook of Logic in Computer Science*, volume 2, pp. 1–112. Oxford University Press, 1992.

[L.G90] L.Gordeev. Generalizations of the kruskal-friedman theorems. *Journal of Symbolic Logic*, Vol. 55, No. 1, pp. 157–181, 1990.

[L.P89] L.Puel. Using unavoidable set of trees to generalize kruskal's theorem. *Journal of Symbolic Computation*, Vol. 8, pp. 335–382, 1989.

[MA92] M.Kurihara and A.Ohuchi. Modularity of simple termination of term rewriting systems with shared constructors. *Theoretical Computer Science*, Vol. 103, pp. 273–282, 1992.

[M.R87] M.Rusinowitch. Path of subterms ordering and recursive decomposition ordering revisited. *Journal of Symbolic Computation*, Vol. 3, pp. 117–131, 1987.

[N.D82] N.Dershowitz. Ordering for term-rewriting systems. *Theoretical Computer Science*, Vol. 17, pp. 279–301, 1982.

[N.D87] N.Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, Vol. 3, pp. 69–116, 1987.

[P.L92] P.Lescanne. Well rewrite orderings and well quasi-orderings. *Journal of Symbolic Computation*, Vol. 14, pp. 419–435, 1992.

[S.G85] S.G.Simpson. Nonprovability of certain combinatorial properties of finite trees. In L.A.Harrington, editor, *Harvey Friedman's research on the foundation of mathematics*, pp. 87–117. Elsevier, 1985.

[Y.T87] Y.Toyama. Counterexamples to termination for the direct sum of term rewriting systems. *Inform. Process. Lett.*, Vol. 25, pp. 141–143, 1987.