Böhm-Like Trees for Term Rewriting Systems

Jeroen Ketema

Department of Computer Science
Faculty of Sciences, Vrije Universiteit Amsterdam
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands

Abstract. In this paper we define Böhm-like trees for term rewriting systems (TRSs). The definition is based on the similarities between the Böhm trees, the Lévy-Longo trees, and the Berarducci trees. That is, the similarities between the Böhm-like trees of the λ-calculus. Given a term t a tree partially represents the root-stable part of t as created in each maximal fair reduction of t. In addition to defining Böhm-like trees for TRSs we define a subclass of Böhm-like trees whose members are monotone and continuous.

1 Introduction

In the theory of the λ-calculus there occur three very similar trees. These are the Böhm trees [1], the Lévy-Longo trees or lazy trees [2], and the Berarducci trees [3]. We call these trees the *Böhm-like trees*. In this paper we define Böhm-like trees for term rewriting system (TRSs). We also define a subclass of Böhm-like trees whose members are monotone and continuous.

The definition of Böhm-like trees for TRSs is based on the similarities between the Böhm-like trees of the λ-calculus. Given a term t a tree partially represents the root-stable part of t as created in each maximal fair reduction of t. Maximal means it is either a reduction to normal form or an infinite reduction. Fair means that every redex occurring in the reduction is eventually contracted.

The actual part as represented by a particular Böhm-like tree depends on the definition of that tree. In the λ-calculus, Böhm trees represent subterms in head normal form, Lévy-Longo trees represent subterms in weak head normal form, and Berarducci trees represent all root-stable subterms.

A root-stable part and a Böhm-like tree can become infinitely large in a maximal reduction. For example, if Y denotes a λ-term that behaves as a fixed-point combinator, then

$$Y(\lambda xy.x) \to_\beta^* \lambda y_1.Y(\lambda xy.x) \to^* \lambda y_1.\lambda y_2.Y(\lambda xy.x) \to_\beta^* \dots$$

and $\lambda y_1.\lambda y_2.\lambda y_3.\dots$ is the Lévy-Longo tree of $Y(\lambda xy.x)$. It is also the Berarducci tree of $Y(\lambda xy.x)$.

Construction. To obtain a partial representation of the root-stable part of a term t, as created in each maximal fair reduction, we construct partial representations of the root-stable parts as created in each finite reduction of t. That is, we construct partial representations of the root-stable parts of the final terms. If we construct representations for final terms of increasingly longer reductions, then in the limit we get a partial representation of the root-stable part of t as created in each maximal fair reduction.

Approaches. There are three approaches to formalising the above limit process. We discuss each of these in turn. The differences between the approaches originate from the different ways in which they represent trees.

Ideal Completion. In this approach unspecified subterms and a partial order on terms are defined first. Then, employing the partial order, trees are defined by means of ideal completion. That is, trees are represented by ideals. The finite and infinite ideals represent respectively the finite and infinite trees. Constructing the partial representation of the root-stable part as created in a finite reduction is done with the help of functions. These functions are called a direct approximant functions. Given a final term of a finite reduction a direct approximant function strips out subterms, leaving them unspecified. At least the non-root-stable subterms are stripped out. The exact definition of a direct approximant function depends on the particular Böhm-like tree [2, 4–6].

Partial Functions. In this approach trees are represented as partial functions from the set of positions to the union of the signature and the variables. The partial functions with a finite and infinite domain represent respectively the finite and infinite trees. Given a term t, the symbol that occurs at a certain position in a Böhm-like tree of t is acquired by recursively reducing t and the subterms of the reduct of t until they are in head normal form, in weak head normal form, or root-stable, depending on the particular tree [1].

Metric Completion. In this approach a metric on terms is defined first. Then, trees are defined by means of metric completion of the set of terms. The terms and the elements created by metric completion represent respectively the finite and infinite trees. The Böhm-like tree of a term is obtained by means of infinitary rewriting in a transfinitely confluent version of the λ-calculus. Rewrite rules of the form $t \to \bot$ are used to obtain transfinite confluence. The actual terms t that occur in the rewrite rules $t \to \bot$ depend on the particular Böhm-like tree [3, 7].

Current Approach. In this paper we use ideal completion to define Böhm-like trees for TRSs. However, to keep the discussion simple

we consider only confluent left-linear TRSs.

Considering non-confluent TRSs at least requires additional clauses in Definition 5.1, as Blom [8] and Ariola and Blom [9] show.

Related Work. The related work can be divided into three categories. First, using ideal completion Boudol [10] and Ariola [11] already defined one particular Böhm-like tree. We discuss this tree in Example 6.11.

Second, Kennaway, Van Oostrom, and De Vries [12] define Böhm-like trees for TRSs on a similar level of abstraction as we do. They use metric completion and infinitary rewriting. To obtain transfinite confluence they formulate sufficient conditions on the terms that may occur in the rewrite rules $t \to \bot$. A comparison of their sufficient conditions and our approach is non-trivial and outside the scope of this paper.

Third, Boudol [10], Blom [8], and Ariola and Blom [9] use ideal completion to define Böhm-like trees that are more abstract than the ones defined here. In their approaches, as we further explain in Sect. 5, the range of the direct approximant functions no longer need to be terms. Their approaches offer excellent frameworks for studying the most abstract properties shared between Böhm-like trees. However, their trees no longer represent the root-stable part of a term as created in each maximal fair reduction. In addition, their direct approximant functions cannot be restricted by relating the domain and range of the functions with the help of partial order on terms. We use such relations when defining the subclass of Böhm-like trees that is monotone and continuous.

Overview. In the rest of this paper we proceed as follows. In Sect. 2 we give some preliminary definitions. Then, in Sect. 3, we define unspecified subterms and the related partial order. In Sect. 4 we define trees, and in Sect. 5 we give a definition of Böhm-like trees. After this we consider a subclass of computable direct approximant functions. The Böhm-like trees based on these direct approximant functions are monotone and continuous. We give the definition of the subclass in Sect. 6, and in Sect. 7 we prove that the trees are monotone and continuous. In Sect. 8, the final section, we give some possible directions for further research.

2 Preliminaries

Most of the notation and concepts we use in this paper correspond to that in the books by Baader and Nipkow [13] and Stoltenberg-Hansen, Lindström, and Griffor [14]. In this section we summarise the most relevant notation and concepts.

Given a signature Σ and a set of variables X, we denote by Σ_n the subset of Σ whose elements have arity n. By $\mathcal{T}er(\Sigma, X)$ we denote the set of terms over Σ and X. We call a term $t \in \mathcal{T}er(\Sigma, X)$ *linear* if each variable from X occurs at most once in t.

If $t \in \mathcal{T}er(\Sigma, X)$, then $\mathcal{P}os(t)$ denotes the set of positions of t. The positions have an associated *prefix order*. We say that p is a *prefix* of q, denoted $p \leq q$, if there exists a position r such that $p \cdot r = q$. Here, the symbol $\cdot$ denotes the *concatenation* of positions and r may be the *empty position* ϵ. We call the positions p and q *parallel* if neither $p \leq q$ nor $q \leq p$.

We denote the subterm of a term t at position $p \in \mathcal{P}os(t)$ by $t|_p$. The replacement of a subterm at position p in t by a term s is denoted $t[s]_p$.

Given a term t and a substitution σ, we denote the application of σ to t by $\sigma(t)$. We also use notation like $t[x := t_1; y := t_2]$. In this case we have a substitution that replaces x by t_1 and y by t_2.

By $\mathcal{R} = (\Sigma, R)$ we denote a TRS over a signature Σ and with the set of rewrite rules R. The elements of R are denoted $l \to r$, where $l, r \in \mathcal{T}er(\Sigma, X)$. We call $\mathcal{R}$ *left-linear*, if the left-hand sides of all its rewrite rules are linear. The rewrite relation defined by R is denoted $\to$. Its reflexive and transitive-reflexive closures are respectively denoted $\to^=$ and $\to^*$.

A TRS $\mathcal{R}$ is *subcommutative*, if for every $s \to t_1$ and $s \to t_2$ there exists a u such that $t_1 \to^= u$ and $t_2 \to^= u$. Moreover, $\mathcal{R}$ is *confluent*, if for every $s \to^* t_1$ and $s \to^* t_2$ there exists a u such that $t_1 \to^* u$ and $t_2 \to^* u$. A term t is in *normal form* with respect to $\mathcal{R}$ if no redex occurs in t. The TRS $\mathcal{R}$ is *terminating* if all reductions are finite.

We call a term t in a TRS *root-stable* if we cannot rewrite t to a term which is a redex. We call a subterm $t|_p$ with $p \in \mathcal{P}os(t)$ root-stable if for all $q \leq p$ the term $t|_q$ is root-stable.

By $\mathcal{P} = (P, \sqsubseteq)$ we denote a partial order $\sqsubseteq$ over a set P. If $Q \subseteq P$, then Q is *consistent* if there exists a $p \in P$ such that for all $q \in Q$ we have $q \sqsubseteq p$. If Q has a least upper bound, then we denote it by $\bigsqcup Q$.

Given a partial order $\mathcal{P} = (P, \sqsubseteq)$, we call a non-empty set $D \subseteq P$ *directed*, if for all $p, q \in D$ there exists an $r \in D$ such that $p \sqsubseteq r$ and $q \sqsubseteq r$. Moreover, we call a non-empty set $D \subseteq P$ *downward closed*, if for all $p \sqsubseteq q$ with $p \in P$ and $q \in D$ we have $p \in D$.

A partial order $\mathcal{P} = (P, \sqsubseteq)$ is a *conditional upper semi-lattice with least element (cusl)*, if P has a least element and if every consistent subset of P has a least upper bound. A set $I \subseteq P$ is an *ideal*, if it is downward closed and if every $\{p, q\} \subseteq I$ is consistent and has a least upper bound in I. An ideal is called finite if it has finite cardinality, otherwise it is called infinite.

For every directed set $D \subseteq P$ in a cusl $\mathcal{P} = (P, \sqsubseteq)$ we can define an ideal, denoted $\downarrow D$, called the *downward closure* of D

$$\downarrow D = \{p \in P \mid p \sqsubseteq q \text{ for some } q \in D\}\,.$$

Moreover, we have that $\mathcal{P}^\infty = (P^\infty, \subseteq)$ is a partial order. Here, P^∞ denotes $\{I \subseteq P \mid I \text{ is an ideal of } \mathcal{P}\}$ and $\subseteq$ denotes subset inclusion. The partial order $\mathcal{P}^\infty$ is called the *ideal completion* of $\mathcal{P}$.

3 Partial Terms

Let Σ be signature and X a set of variables. To represent unspecified subterms we extend the signature with a constant $\perp$ which neither occurs in Σ nor in X. The *unspecified subterms* are defined as those subterms that are equal to $\perp$.

We call the set of terms over the signature $\Sigma \cup \{\perp\}$ the set of *partial terms*. We denote the set by $\mathcal{T}er(\Sigma_\perp, X)$. We leave out the adjective partial when it is obvious from the context.

Given a TRS $\mathcal{R} = (\Sigma, R)$ we can define the TRS $\mathcal{S} = (\Sigma \cup \{\perp\}, R)$. The definition of $\mathcal{S}$ is sound, as $\Sigma \subseteq \Sigma \cup \{\perp\}$. Moreover, $\mathcal{S}$ has the same confluence and termination properties as $\mathcal{R}$, as we can consider $\perp$ to be a variable which we have singled out.

With the help of $\perp$ we can define a partial order on terms, called the prefix order. We can also define a strict partial order, called the strict prefix order.

Definition 3.1. *Let Σ be a signature and X a set of variables.*

1. *The* prefix order *on $\mathcal{T}er(\Sigma_\perp, X)$, denoted $\preccurlyeq$, is the smallest binary relation such that*
   (a) *$x \preccurlyeq x$ for all $x \in X$,*
   (b) *$\perp \preccurlyeq t$ for all $t \in \mathcal{T}er(\Sigma_\perp, X)$, and*
   (c) *$f(s_1, \ldots, s_n) \preccurlyeq f(t_1, \ldots, t_n)$ for all $f \in \Sigma_n$ and $s_i \preccurlyeq t_i$ with $1 \leq i \leq n$.*
2. *The* strict prefix order *on $\mathcal{T}er(\Sigma_\perp, X)$, denoted $\prec$, is the smallest binary relation such that for all $s, t \in \mathcal{T}er(\Sigma_\perp, X)$*

$$s \prec t \text{ iff } s \preccurlyeq t \text{ and } s \neq t\,.$$

If $s \preccurlyeq t$, then we call s a *prefix* of t. Moreover, if $s \prec t$, then we call s a *strict prefix* of t. The term s is a prefix of the term t if either s and t are equal or if there exist unspecified subterms in s which are specified in t but not the other way around. See Fig. 1 for a graphical representation.

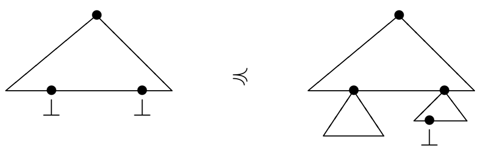

Fig. 1. The prefix order on $\mathcal{T}er(\Sigma_\perp, X)$

By induction on the structure of terms it follows that $\mathcal{PT} = (\mathcal{T}er(\Sigma_\perp, X), \preccurlyeq)$ and $\mathcal{SPT} = (\mathcal{T}er(\Sigma_\perp, X), \prec)$ are respectively a partial order and a strict partial order. The pair $\mathcal{PT}$ is in fact a cusl. The existence of a least element, the constant $\perp$, follows by the second clause of the prefix order and the anti-symmetry of partial orders. By the same facts and by induction on the structure of terms it follows that every consistent set of terms has a least upper bound.

We have the following relations between the prefix orders and the positions of terms.

Lemma 3.2. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, X)$.*

1. *For all $s \preccurlyeq t$*
   – *$\mathcal{P}os(s) \subseteq \mathcal{P}os(t)$, and*
   – *for all $p \in \mathcal{P}os(s)$ such that $s|_p \neq \perp$, the root symbol of $s|_p$ is equal to the root symbol of $t|_p$.*
2. *For all $s \prec t$, there exist $p \in \mathcal{P}os(s)$ such that $s|_p = \perp$ and $t|_p \neq \perp$.*

Proof. By induction on the structure of terms. $\square$

Using the previous lemma we prove well-foundedness of the strict prefix order.

Proposition 3.3. *The strict prefix order on $\mathcal{T}er(\Sigma_\perp, X)$ is well-founded.*

Proof. Let $s, t \in \mathcal{T}er(\Sigma_\perp, X)$ with $s \prec t$. From Lemma 3.2 it follows that

$$\#\{p \mid p \in \mathcal{P}os(s),\ s|_p \neq \perp\} < \#\{p \mid p \in \mathcal{P}os(t),\ t|_p \neq \perp\}\,,$$

where $\#S$ denotes the cardinality of S. Hence, as $<$ is a well-founded order on the natural numbers, the result follows. □

We can extend the prefix order to substitutions by means of a point-wise definition. That is, given substitutions σ and τ

$$\sigma \preccurlyeq \tau \text{ iff } \sigma(x) \preccurlyeq \tau(x) \text{ for all } x \in X\,.$$

Using this definition we can also extend the strict prefix order to substitutions

$$\sigma \prec \tau \text{ iff } \sigma \preccurlyeq \tau \text{ and } \sigma(x) \prec \tau(x) \text{ for some } x \in X\,.$$

Thus, for all variables we must have $\sigma(x) \preccurlyeq \tau(x)$ and for at least one variable we must also have $\sigma(x) \prec \tau(x)$.

The extensions of the prefix order and the strict prefix order to substitutions are again respectively a partial order and a strict partial order. This follows easily from their definitions and the fact that the prefix order and the strict prefix order on terms are respectively a partial order and a strict partial order.

The following property holds with respect to the extension of the prefix order to substitutions. The property plays an essential rôle in Sect. 6.

Lemma 3.4. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, X)$ such that t is linear. If $s \preccurlyeq \tau(t)$ for some substitution τ, then there exists an $s' \in \mathcal{T}er(\Sigma_\perp, X)$ and a substitution σ' such that $s = \sigma'(s')$, $s' \preccurlyeq t$, $\sigma' \preccurlyeq \tau$, and s' linear.*

Proof. Suppose $s \preccurlyeq \tau(t)$ for some substitution τ. We prove the result by induction on the number of positions $p \in \mathcal{P}os(s)$ such that $s|_p = \perp$ and $t|_p \neq \perp$.

Base Case. There are no positions p such that $s|_p = \perp$ and $t|_p \neq \perp$. Hence, $s = \tau(t)$ and the result follows by defining $s' = t$ and $\sigma' = \tau$.

Induction Step. Suppose the result holds for some number of positions $n \geq 0$. Let us prove it holds for $n + 1$ positions.

As $n + 1 > 0$, there exists a position $p \in \mathcal{P}os(s)$ such that $s|_p = \perp$ and $\tau(t)|_p \neq \perp$. With respect to p there are two possibilities

1. p is a non-variable position of t, or
2. there exists a variable position q of t such that $p = q \cdot r$.

In the first case define

$$\begin{aligned} t' &= t[\perp]_p \\ \tau'(x) &= \tau(x) \text{ for all } x \in X. \end{aligned}$$

In the second case define

$$t' = t$$
$$\tau'(x) = \begin{cases} \tau(x)[\bot]_r & \text{if } x = t|_q \\ \tau(x) & \text{otherwise.} \end{cases}$$

In both cases $t' \preccurlyeq t$, $\tau' \preccurlyeq \tau$ and t' linear. Moreover, as $s|_p = \bot$ and $\tau(t)|_p \neq \bot$, it follows that $s \preccurlyeq \tau'(t') \prec \tau(t)$ and that p is the only position such that $\tau'(t')|_p = \bot$ and $\tau(t)|_p \neq \bot$. Consequently, the number of positions p with $s|_p = \bot$ and $\tau'(t')|_p \neq \bot$ is n, and by the induction hypothesis it follows that there exist an s' and σ' such that $s = \sigma'(s')$, $s' \preccurlyeq t'$, $\sigma' \preccurlyeq \tau'$, and s' linear. The actual result follows by transitivity of the prefix orders on terms and substitutions. □

We conclude this section with two remarks regarding the previous lemma.

Remark 3.5. If the position p as used in the induction step is a variable position of the term t, then there is in fact more than one way to construct t' and τ'. Consider, for example, $s = f(\bot, a)$, $t = f(x, y)$, and $\tau = [x := a; y := a]$. Following the proof of the lemma we have

$$f(x, y)[x := \bot; y := a] = f(\bot, a) \preccurlyeq f(a, a) = f(x, y)[x := a; y := a]\,.$$

However, we also have

$$f(\bot, y)[x := a; y := a] = f(\bot, a) \preccurlyeq f(a, a) = f(x, y)[x := a; y := a]\,.$$

That is, in the first case $t' = f(x, y)$ and $\tau' = [x := \bot; y := a]$ and in the second case $t' = f(\bot, y)$ and $\tau' = [x := a; y := a]$.

Remark 3.6. If t is not assumed to be linear, it is in general not possible to prove the lemma. Consider, for example, $s = f(g(\bot), g(a))$, $t = f(x, x)$, and $\tau = [x := g(a)]$. Although we have

$$f(g(\bot), g(a)) \preccurlyeq f(g(a), g(a)) = f(x, x)[x := g(a)]\,,$$

there does not exist a substitution σ' such that $\sigma'(f(x, x)) = f(g(\bot), g(a))$. The first argument of s is not equal to its second argument.

4 Trees

We define the set of trees by means of ideal completion.

Definition 4.1. *Let Σ be a signature and X a set of variables. The set of* trees, *denoted $\mathcal{T}^\infty(\Sigma_\bot, X)$, is defined by*

$$\mathcal{T}^\infty(\Sigma_\bot, X) = \{I \subseteq \mathcal{T}er(\Sigma_\bot, X) \mid I \text{ is an ideal of } \mathcal{PT}\}\,.$$

In this definition the finite and infinite ideals represent respectively the *finite trees* and *infinite trees*. We do not explain ideal completion any further. This has been done elsewhere [14].

The following three concepts are related to trees.

Definition 4.2. *Let $S, T \in \mathcal{T}^\infty(\Sigma_\perp, X)$. Define*

Prefix Order $S \preccurlyeq T$ *iff for all $s \in S$ there exist $t \in T$ such that $s \preccurlyeq t$,*
Positions $\mathcal{P}os(T) = \bigcup\{\mathcal{P}os(t) \mid t \in T\}$*, and*
Subtree $T|_p = \{t|_p \mid t \in T,\ p \in \mathcal{P}os(t)\}$ *if $p \in \mathcal{P}os(T)$.*

Two remarks are in order with respect to this definition. First, as trees are ideals, the prefix order is in fact subset inclusion. Hence, the least upper bound of a consistent set of trees is its union. Second, as follows immediately from its definition, $T|_p$ is an ideal, and it is finite when T is finite.

We can clarify the chosen terminology with the help an isomorphism ι from $\mathcal{T}er(\Sigma_\perp, X)$ to the finite ideals of $\mathcal{T}^\infty(\Sigma_\perp, X)$. Given a term t, the isomorphism is defined by

$$\iota(t) = \downarrow\{t\} = \{s \mid s \preccurlyeq t\}\,.$$

The set $\iota(t)$ is finite. This follows from the definition of the prefix order and from the fact that t has a finite number of symbols. The set $\iota(t)$ is also an ideal. This follows by the definition of downward closure.

The inverse of ι assigns to each finite ideal I its least upper bound. That is,

$$\iota^{-1}(I) = \bigsqcup I\,.$$

The existence of the least upper bound of I follows by the definition of finite ideals. By this fact and the facts about $\iota(t)$ it follows easily that ι actually is an isomorphism. Hence, each term corresponds to a finite ideal and vice versa. As we can view every term as a finite tree, we also call a finite ideal a finite tree.

The following observations relate the concepts from Definition 4.2 with the prefix order, the set of positions, and the replacement of a subterm, as defined in the preliminaries. We assume that $s, t \in \mathcal{T}er(\Sigma_\perp, X)$ and that S and T are finite ideals of $\mathcal{T}^\infty(\Sigma_\perp, X)$.

$$\begin{array}{rclcrcl}
s \preccurlyeq t & \text{iff} & \iota(s) \preccurlyeq \iota(t) & \qquad & S \preccurlyeq T & \text{iff} & \iota^{-1}(S) \preccurlyeq \iota^{-1}(T) \\
\mathcal{P}os(t) & = & \mathcal{P}os(\iota(t)) & & \mathcal{P}os(T) & = & \mathcal{P}os(\iota^{-1}(T)) \\
\iota(t|_p) & = & \iota(t)|_p & & \iota^{-1}(T|_p) & = & \iota^{-1}(T)|_p
\end{array}$$

5 Böhm-Like Trees

A Böhm-like tree of a term t partially represents the root-stable part of t as created in each maximal fair reduction of t. To obtain a Böhm-like tree of t we construct partial representations of the root-stable parts of the final terms of all finite reductions. This is done with a direct approximant function. The definition of such a function depends on the particular Böhm-like tree. However, all direct approximant functions must satisfy the following definition. It summarises the properties shared between the direct approximants functions defined in earlier papers [2, 4–6, 11].

Definition 5.1. *Let $\mathcal{R} = (\Sigma, R)$ be a TRS. A* direct approximant function *of $\mathcal{R}$ is a function $\omega : \mathcal{T}er(\Sigma_\perp, X) \to \mathcal{T}er(\Sigma_\perp, X)$, such that for all $s, t \in \mathcal{T}er(\Sigma_\perp, X)$ and substitutions σ*

1. $\omega(t) \preccurlyeq t$,
2. *if* $t|_p = \sigma(l)$, *then* $\omega(t) \preccurlyeq t[\bot]_p$ *for all* $p \in \mathcal{P}os(t)$ *and* $l \to r \in R$, *and*
3. *if* $s \to t$, *then* $\omega(s) \preccurlyeq \omega(t)$.

In the remainder of this section we assume $\mathcal{R} = (\Sigma, R)$ is a confluent left-linear TRS and ω is a direct approximant function of $\mathcal{R}$.

Given a term t, we call $\omega(t)$ the *direct approximant* of t. Note that by the first and second clause of Definition 5.1 a direct approximant is in normal form with respect to $\mathcal{R}$.

The first clause of Definition 5.1 expresses that a direct approximant of a term is a prefix of that term. Note that the root-stable part of a term, or any of its prefixes, is such a prefix. The first clause is a consequence of the second clause for terms not in normal form.

The second and third clause of Definition 5.1 are motivated by the following lemma. It expresses that a direct approximant only provides information on the root-stable part of a term.

Lemma 5.2. *Let* $t \in \mathcal{T}er(\Sigma_\bot, X)$ *and* $p \in \mathcal{P}os(t)$. *If* $t|_p$ *is not a root-stable subterm of* t, *then there exists a* $q \le p$ *such that* $q \in \mathcal{P}os(\omega(t))$ *and* $\omega(t)|_q = \bot$.

Proof. This follows immediately from the definition of root-stable subterms and the second and third clause of Definition 5.1. ☐

We are now almost ready to define Böhm-like trees. However, we first need to define the notion of auxiliary set. An auxiliary set of a term t consists of the direct approximants of all the reducts of t.

Definition 5.3. *If* $t \in \mathcal{T}er(\Sigma_\bot, X)$, *then its* auxiliary set, *denoted* $\mathcal{A}(t)$, *is defined by*

$$\mathcal{A}(t) = \{\omega(s) \mid t \to^* s\}\,.$$

Auxiliary sets have the following property.

Lemma 5.4. *Let* $t \in \mathcal{T}er(\Sigma_\bot, X)$. *The set* $\mathcal{A}(t)$ *is directed.*

Proof. The set $\mathcal{A}(t)$ is non-empty, as follows from the fact that $\omega(t) \in \mathcal{A}(t)$. Moreover, for all $s_1, s_2 \in \mathcal{A}(t)$ there exist an $r \in \mathcal{A}(t)$ such that $s_1 \preccurlyeq r$ and $s_2 \preccurlyeq r$, as follows from the third clause of Definition 5.1 and the assumption that all considered TRSs are confluent. ☐

The set $\mathcal{A}(t)$ is not necessarily a tree. Consider, for example, the TRS $\mathcal{R} = (\{c\}, \emptyset)$ with c a constant. Since there are no reduction rules, the identity function on $\mathcal{T}er(\{c\}_\bot, X)$ is a direct approximant function. Hence, we have $\mathcal{A}(c) = \{c\}$. This is not a tree, as $\bot \notin \{c\}$. However, as $\mathcal{A}(t)$ is directed and as trees are ideals we can obtain a tree by closing $\mathcal{A}(t)$ downward. This leads to the following definition of Böhm-like trees.

Definition 5.5. *If* $t \in \mathcal{T}er(\Sigma_\bot, X)$, *then its* Böhm-like tree, *denoted* $BLT(t)$, *is defined by*

$$BLT(t) = \downarrow \mathcal{A}(t)\,.$$

We have for each t that $\downarrow\mathcal{A}(t)$ exists and is unique. Hence, BLT is a function from $\mathcal{T}er(\Sigma_\perp, X)$ to $\mathcal{T}^\infty(\Sigma_\perp, X)$. By Lemma 5.2 and the fact that root-stability is preserved under reduction, Böhm-like trees only provide information on root-stable parts.

We now give two examples of direct approximant functions and Böhm-like trees.

Example 5.6 (Trivial Trees). Given a term t, its trivial direct approximant is defined by $\omega_{\mathrm{T}}(t) = \perp$.

The three clauses of Definition 5.1 hold trivially. As we have for all $t \to^* s$ that $\omega_{\mathrm{T}}(s) = \perp$, it follows that $\mathrm{BLT}(t) = \mathcal{A}(t) = \{\perp\}$. Note that ω_{T} is minimal in the sense that it does not provide any information on root-stable subterms.

Example 5.7 (Berarducci-Like Trees). Given a term t, its Berarducci-like direct approximant ω_{BeL} replaces precisely all non-root-stable subterms of t by $\perp$.

Again, the three clauses of Definition 5.1 hold trivially. Note that ω_{BeL} is maximal in the sense that it preserves all root-stable subterms. Unfortunately, as root-stability is undecidable, ω_{BeL} is in general not computable.

Berarducci-like trees are modelled after the Berarducci trees from the λ-calculus [3]. The direct approximant function associated with the Berarducci trees also replaces precisely all non-root-stable subterms by $\perp$.

To make the Berarducci-like trees more concrete let us consider combinatory logic (CL) with the combinators S, K, and I and the usual reduction rules.

The following trees are Berarducci-like trees for CL.

$$\begin{aligned} \mathrm{BLT}(K\perp) &= \{\perp, \perp\perp, K\perp\} \\ \mathrm{BLT}(YK) &= \{\perp, \perp\perp, K\perp, \perp(\perp\perp), \ldots\} \\ \mathrm{BLT}(SII(SII)) &= \{\perp\} \end{aligned}$$

The subterm Y in the second tree denotes a term that behaves as a fixed-point combinator. In the case of the last tree note that for every $SII(SII) \to^* t$ we have $t \to^* SII(SII)$. Hence, no reduct of $SII(SII)$ is root-stable.

We end this section with a proof that Böhm-like trees are preserved under rewriting and by discussing some related work.

Proposition 5.8. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, X)$. If $s \to^* t$, then $BLT(s) = BLT(t)$.*

Proof. Suppose $s \to^* t$. We prove $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$ and $\mathrm{BLT}(t) \preccurlyeq \mathrm{BLT}(s)$. The result follows from the observation that the prefix order on trees is in fact subset inclusion.

By the definition of Böhm-like trees there exists for every $t'' \in \mathrm{BLT}(s)$ a term t' such that $s \to^* t'$ and $t'' \preccurlyeq \omega(t')$. As we assume that every TRS is confluent, there exists an r such that $t \to^* r$ and $t' \to^* r$. Thus, $\omega(r) \in \mathcal{A}(t) \subseteq \mathrm{BLT}(t)$. Moreover, by the third clause of Definition, 5.1 $\omega(t') \preccurlyeq \omega(r)$. Hence, $t'' \preccurlyeq \omega(r)$ and $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$.

As every reduct of t is a reduct of s, we have $\mathcal{A}(t) \subseteq \mathcal{A}(s)$. By the definition of downward closure $\downarrow\mathcal{A}(t) \subseteq \downarrow\mathcal{A}(s)$. Thus, $\mathrm{BLT}(t) \preccurlyeq \mathrm{BLT}(s)$. $\square$

In the work by Boudol [10], Blom [8], and Ariola and Blom [9] a more abstract approach is taken to defining Böhm-like trees. They use more abstract definitions of direct approximant functions.

Boudol [10] only requires of the range of the direct approximant function that its is an algebra over $\mathcal{T}er(\Sigma_\perp, X)$. The range does not need to be $\mathcal{T}er(\Sigma_\perp, X)$. In correspondence with this, Boudol drops the first clause of Definition 5.1.

Blom [8] and Ariola and Blom [9] require the domain of the direct approximant function only to be an ARS $\mathcal{A} = (A, \to)$ with a partial order on A. The ARS $\mathcal{A}$ does not need to be confluent. The range of the direct approximant function may be an arbitrary (complete) partial order. In correspondence with this, they drop the first and second clause of Definition 5.1. They also add a new clause to compensate for the fact that $\mathcal{A}$ does not need to be confluent.

6 Direct Approximant TRSs

In this section we define a class of confluent and terminating TRSs, the direct approximant TRSs (ωTRSs). We prove that the function that assigns to each term in such a TRS its unique normal form is a direct approximant function. In the next section we prove that the Böhm-like trees based on ωTRSs are monotone and continuous.

Not every direct approximant function can be defined by means of a confluent and terminating TRS. An example is the function ω_{BeL} from the previous section. This function cannot be defined by means of a TRS, as unique normal forms of confluent and terminating TRSs are always computable, while root-stability and, hence, ω_{BeL} is not.

As in the case of direct approximant functions, the definition of ωTRSs is relative to a given TRS. The definition summarises the properties shared between the TRSs used to define direct approximant functions in earlier papers [6, 10, 11, 15].

Definition 6.1. *Let $\mathcal{R} = (\Sigma, R)$ be a confluent left-linear TRS. A* direct approximant TRS (ωTRS) *of $\mathcal{R}$ is a left-linear TRS $\mathcal{D} = (\Sigma_\perp, D)$, whose rewrite relation, denoted $\to_\omega$, satisfies*

1. *$e = \perp$ for all $d \to_\omega e \in D$,*
2. *$\perp$ is a normal form with respect to $\to_\omega$,*
3. *$t \to_\omega^* \perp$ for all $t \preccurlyeq d$ with $d \to_\omega \perp \in D$ (see Fig. 2), and*
4. *$l \to_\omega^* \perp$ for all $l \to r \in R$.*

In the remainder of this section we assume $\mathcal{R} = (\Sigma, R)$ is a confluent left-linear TRS and $\mathcal{D} = (\Sigma_\perp, D)$ is a ωTRS of $\mathcal{R}$. We proceed as follows. First, we give an example of a ωTRS. Then, we prove ωTRSs are confluent and terminating using the first, second, and third clause of Definition 6.1. Finally, we prove that the unique normal forms define direct approximants using the third and fourth clause.

$$\begin{tikzcd} t \arrow[d, dotted, "*"] & \preccurlyeq & d \arrow[d, "\omega"] \\ \bot & = & \bot \end{tikzcd}$$

Fig. 2. Definition 6.1.(3)

$$\begin{tikzcd} s \arrow[d, dotted, "*_\omega"] & \preccurlyeq & t \arrow[d, "*_\omega"] \\ s' & \preccurlyeq & t' \end{tikzcd}$$

Fig. 3. Lemma 6.7

$$\begin{tikzcd} s \arrow[rr, "*"] \arrow[d, dotted, "*_\omega"] & & t \arrow[d, "*_\omega"] \\ s' & \preccurlyeq & t' \end{tikzcd}$$

Fig. 4. Lemma 6.8

Example 6.2 (Huet-Lévy ωTRSs). The rewrite rules of the Huet-Lévy ωTRS are all rules of the form $t \to_\omega \bot$ such that $\bot \neq t \preccurlyeq l$ and $l \to r \in R$.

The four clauses of Definition 6.1 follow trivially from the definition of Huet-Lévy ωTRSs. The direct approximant function defined by a Huet-Lévy ωTRS originates from the work by Huet and Lévy [16]. The first formulation as a TRS is by Klop and Middeldorp [15]. The definition of Klop and Middeldorp differs slightly from ours, but equality of the transitive-reflexive closures follows easily with the help of Lemma 3.4.

The Huet-Lévy TRS for CL has no less than 28 rewrite rules. However, using the fact that the third clause of Definition 6.1 is formulated in terms of the transitive-reflexive closure of $\to_\omega$, we can define a ωTRS with the same transitive-reflexive closure but with only four rewrite rules.

$$\begin{array}{rr} Sxyz \to_\omega \bot & Kxy \to_\omega \bot \\ Ix \to_\omega \bot & \bot x \to_\omega \bot \end{array}$$

Hence, the formulation of the third clause of Definition 6.1 enables us to define more "economic" ωTRSs.

To prove confluence of $\mathcal{D}$ we first show that confluence holds for ωTRSs for which the third clause of Definition 6.1 can be strengthened to

$t \to_\omega^= \bot$ *for all* $t \preccurlyeq d$ *with* $d \to_\omega \bot \in D$.

That is, t must rewrite to $\bot$ in at most one step and not just in finitely many steps. We call ωTRSs with this strengthened third clause *single-step ωTRSs* .

Lemma 6.3. *If $\mathcal{E} = (\Sigma_\bot, E)$ is a single-step ωTRS, then $\mathcal{E}$ is confluent.*

Proof. The ωTRS $\mathcal{E} = (\Sigma_\bot, E)$ is subcommutative by the first clause of Definition 6.1 and the single-step assumption. Confluence is implied by subcommutativity [13, Lemma 2.7.4]. □

Using confluence of single-step ωTRSs we can prove confluence of $\mathcal{D}$.

Proposition 6.4. *The ωTRS $\mathcal{D}$ is confluent.*

Proof. Define a TRS $\mathcal{E} = (\Sigma_\bot, E)$, such that $t \to_\omega \bot \in E$ for all $t \in \mathcal{T}er(\Sigma_\bot, X)$ with $\bot \neq t \preccurlyeq d$ and $d \to_\omega \bot \in D$. The TRS $\mathcal{E}$ is a single-step ωTRS, as follows easily from its definition. Moreover, by the definition of $\mathcal{E}$, the transitive-reflexive closure of $\mathcal{D}$ and $\mathcal{E}$ are equal. Hence, $\mathcal{D}$ is confluent by Lemma 6.3. □

To prove termination of $\mathcal{D}$ we need the following lemma with respect to the rewrite relation of $\mathcal{D}$.

Lemma 6.5. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, X)$. If $s \to_\omega t$, then $t \prec s$.*

Proof. By the first clause of Definition 6.1 a reduction step $s \to_\omega t$ is a replacement of a subterm s' at a position p in s by $\perp$. As $\perp \prec s'$, we have $t = s[\perp]_p \prec s[s']_p = s$. □

We can now prove termination.

Proposition 6.6. *The ωTRS $\mathcal{D}$ is terminating.*

Proof. By Lemma 6.5 and Proposition 3.3. □

By Propositions 6.4 and 6.6 each term t in $\mathcal{D}$ has a unique normal form. We denote this unique normal form by $\omega(t)$. We now prove that ω defines a direct approximant function. In order to do this, we first prove three lemmas.

Lemma 6.7. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\perp, X)$. If $s \preccurlyeq t$ and $t \to_\omega^* t'$, then there exists an $s' \in \mathcal{T}er(\Sigma_\perp, X)$ such that $s' \preccurlyeq t'$ and $s \to_\omega^* s'$ (see Fig. 3).*

Proof. We give a proof for the case $t \to_\omega t'$. The result follows by induction on the length of $t \to_\omega^* t'$.

Suppose the redex contracted in $t \to_\omega t'$ occurs at position p. There are two cases to consider depending on the occurrence of p in s.

The position p does not occur in s. By the definition of the prefix order there exists a $q \leq p$ such that $s|_q = \perp$. Define $s' = s$. As $t \to_\omega t'$ replaces the subterm at position p by $\perp$ we have by $s|_q = \perp$ and $q \leq p$ that $s \preccurlyeq t'$. Moreover, $s \to_\omega^* s = s'$.

The position p occurs in s. In this case, $s|_p \preccurlyeq t|_p$. As $t|_p$ is a redex, we have by Lemma 3.4 and the third clause of Definition 6.1 that $s|_p \to_\omega^* \perp = t'|_p$. Define $s' = s[\perp]_p$. As $t' = t[\perp]_p$, we have $s' \preccurlyeq t'$. Moreover, as $s|_p \to_\omega^* \perp$, we have $s \to_\omega^* s'$. □

Lemma 6.8. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\perp, X)$. If $s \to^* t$ and $t \to_\omega^* t'$, then there exists an $s' \in \mathcal{T}er(\Sigma_\perp, X)$ such that $s \to_\omega^* s'$ and $s' \preccurlyeq t'$ (see Fig. 4).*

Proof. We give a proof for the case $s \to t$. The result follows by induction on the length of $s \to^* t$.

Suppose the redex contracted in $s \to t$ occurs at position p. As $s[\perp]_p \preccurlyeq t$, there exists by Lemma 6.7 an s' such that $s' \preccurlyeq t'$ and $s[\perp]_p \to_\omega^* s'$. Moreover, $s \to_\omega^* s'$, because by the fourth clause of Definition 6.1 $s \to_\omega^* s[\perp]_p$. □

Lemma 6.9. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, X)$. The following properties hold*

1. $\omega(t) \preccurlyeq t$,
2. $\omega(t) = \omega(t[\omega(t|_p)]_p)$ *for all* $p \in \mathcal{P}os(t)$,
3. $\omega(\omega(t)) = \omega(t)$,

4. $\omega(s) \preccurlyeq \omega(t)$ *if* $s \preccurlyeq t$*, and*
5. $\omega(s) \preccurlyeq \omega(t)$ *if* $s \to t$*.*

Proof. 1. As $\omega(t)$ is the unique normal form of t, we have $t \to_\omega^* \omega(t)$. The result follows by repeated application of Lemma 6.5.
2. For every $t|_p \to_\omega^* s$ we have $t = t[t|_p]_p \to_\omega^* t[s]_p$. Hence, as $t|_p \to_\omega^* \omega(t|_p)$, the result follows by confluence of ωTRSs.
3. By the second clause of the current lemma with $p = \epsilon$.
4. As $t \to_\omega^* \omega(t)$, there exists by Lemma 6.7 an s' such that $s' \preccurlyeq \omega(t)$. Moreover, by confluence of ωTRSs $\omega(s') = \omega(s)$ and by the first clause of the current lemma $\omega(s') \preccurlyeq s'$. Hence, by transitivity of the prefix order $\omega(s) \preccurlyeq \omega(t)$.
5. Analogous to the fourth clause of the current lemma using Lemma 6.8 instead of Lemma 6.7. □

We can now prove the following theorem.

Theorem 6.10. *The function* $\omega : \mathcal{T}er(\Sigma_\perp, X) \to \mathcal{T}er(\Sigma_\perp, X)$ *which assigns to each term its unique normal form with respect to* $\mathcal{D}$ *is a direct approximant function.*

Proof. The first clause of Definition 5.1 follows from Lemma 6.9.(1). The second clause follows from the fourth clause of Definition 6.1 and the fact that ωTRSs are confluent. The third clause follows from Lemma 6.9.(5). □

We now know that each ωTRS defines direct approximant function. Hence, it also defines a Böhm-like tree.

Example 6.11 (Huet-Lévy Trees). The Huet-Lévy ωTRS of Definition 6.2 defines the *Huet-Lévy tree.*

The Huet-Lévy tree is the Böhm-like tree already defined by Boudol [10] and Ariola [11].

Huet-Lévy trees provide more information than the trivial trees, but less than the Berarducci-like trees. For example, given the TRS with the single rewrite rule $f(a) \to b$ we have the following trees.

$$\begin{array}{rr} \mathrm{BLT}_{\mathrm{T}}(f(\perp)) = \{\perp\} & \mathrm{BLT}_{\mathrm{T}}(f(a)) = \{\perp\} \\ \mathrm{BLT}_{\mathrm{HL}}(f(\perp)) = \{\perp\} & \mathrm{BLT}_{\mathrm{HL}}(f(a)) = \{\perp, b\} \\ \mathrm{BLT}_{\mathrm{BeL}}(f(\perp)) = \{\perp, f(\perp)\} & \mathrm{BLT}_{\mathrm{BeL}}(f(a)) = \{\perp, b\} \end{array}$$

7 Monotonicity and Continuity

In this section we prove that a Böhm-like tree whose the direct approximant function can be defined by means of a ωTRS is monotone and continuous. As in the previous section, we assume $\mathcal{R} = (\Sigma, R)$ is an confluent left-linear TRS and $\mathcal{D} = (\Sigma_\perp, D)$ is a ωTRS of $\mathcal{R}$.

Proposition 7.1. *The Böhm-like tree defined by* $\mathcal{D}$ *is a monotone function. That is, for all* $s, t \in \mathcal{T}er(\Sigma_\perp, X)$*, if* $s \preccurlyeq t$*, then* $BLT(s) \preccurlyeq BLT(t)$*.*

Proof. Let $s, t \in \mathcal{T}er(\Sigma_\bot, X)$ such that $s \preccurlyeq t$. Suppose $s'' \in \mathrm{BLT}(s)$. By the definition of $\mathrm{BLT}(s)$ there exists an s' such that $s'' \preccurlyeq \omega(s')$ and $s \to^* s'$. As all assumed TRSs are left-linear, there exists a t' such that $t \to^* t'$ and $s' \preccurlyeq t'$. By Lemma 6.9.(4) we have $\omega(s') \preccurlyeq \omega(t')$. Hence, as $\omega(t') \in \mathrm{BLT}(t)$, we also have $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$. □

Proposition 7.2. *The Böhm-like tree defined by $\mathcal{D}$ is a continuous function. That is, if $t \in \mathcal{T}er(\Sigma_\bot, X)$, then $BLT(t) = \bigsqcup\{BLT(s) \mid s \preccurlyeq t\}$.*

Proof. Let $t \in \mathcal{T}er(\Sigma_\bot, X)$. As $t \preccurlyeq t$, we have $\mathrm{BLT}(t) \in \{\mathrm{BLT}(s) \mid s \preccurlyeq t\}$. Thus, $\mathrm{BLT}(t) \preccurlyeq \bigsqcup\{\mathrm{BLT}(s) \mid s \preccurlyeq t\}$. Moreover, by Proposition 7.1 we have for all $s \preccurlyeq t$ that $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$ and, thus, $\bigsqcup\{\mathrm{BLT}(s) \mid s \preccurlyeq t\} \preccurlyeq \mathrm{BLT}(t)$. Combining both facts, we get the result. □

From the above two propositions we can conclude that the Huet-Lévy trees of the previous section are monotone and continuous. Note that Ariola [11] already proves this.

There exist Böhm-like trees that are not monotone and continuous. Consider, for example, the TRS with the single rewrite rule $f(a) \to b$ and its Berarducci-like tree. Given the terms $f(\bot)$ and $f(a)$ we have that $f(\bot) \preccurlyeq f(a)$, but

$$\mathrm{BLT}(f(\bot)) = \{\bot, f(\bot)\} \not\preccurlyeq \{\bot, b\} = \mathrm{BLT}(f(a))$$

and

$$\mathrm{BLT}(f(a)) = \{\bot, b\} \neq \{\bot, b, f(\bot)\} = \bigcup\{\mathrm{BLT}(s) \mid s \preccurlyeq f(a)\}\,.$$

In fact, the last set is not even a tree.

8 Further Directions

There are at least four interesting directions for further research. First, does precongruence hold for the presented Böhm-like trees, as it does for the Böhm-like trees of the λ-calculus [1–3] and Huet-Lévy trees [11]? That is, suppose $C[\Box]$ is a context and s and t are terms, does it hold that

$$\mathrm{BLT}(C[s]) \preccurlyeq \mathrm{BLT}(C[t]) \text{ if } \mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$$

Second, can we extend Böhm-like trees to higher-order rewriting systems, such that we also cover the Böhm-like trees of the λ-calculus? Third, similar to Berarducci-like trees and the Berarducci trees of the λ-calculus, do Böhm trees [1] and Lévy-Longo trees [2] have a counterpart for TRSs? Fourth, how does the current approach relate to the infinitary rewriting approach [12]?

Acknowledgements. I would like to thank Stefan Blom, Bas Luttik, Jan Willem Klop, Femke van Raamsdonk, Roel de Vrijer, and the anonymous referees for their helpful comments and remarks.

References

1. Barendregt, H.P.: The Lambda Calculus: Its Syntax and Semantics. Second edn. Elsevier Science (1985)
2. Lévy, J.J.: An algebraic interpretation of the $\lambda\beta$K-calculus and the labelled λ-calculus. In Böhm, C., ed.: λ-calculus and Computer Science Theory. Volume 37 of LNCS. Springer-Verlag (1975) 147–165
3. Berarducci, A.: Infinite λ-calculus and non-sensible models. In Ursini, A., Aglianò, P., eds.: Logic and Algebra, Marcel Dekker (1996) 339–378
4. Lévy, J.J.: Réductions correctes et optimales dans le lambda-calcul. PhD thesis, Université de Paris VII (1978)
5. Hyland, M.: A syntactic characterization of the equality in some models for the lambda calculus. Journal of the London Mathematical Society (2) **12** (1976) 361–370
6. Wadsworth, C.P.: The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus. SIAM Journal on Computing **5** (1976) 488–521
7. Kennaway, J.R., Klop, J.W., Sleep, M., de Vries, F.J.: Infinitary lambda calculus. Theoretical Computer Science **175** (1997) 93–125
8. Blom, S.C.C.: Term Graph Rewriting: syntax and semantics. PhD thesis, Vrije Universiteit Amsterdam (2001)
9. Ariola, Z.M., Blom, S.: Skew confluence and the lambda calculus with letrec. Annals of Pure and Applied Logic **117** (2002) 95–168
10. Boudol, G.: Computational semantics of term rewriting systems. In Nivat, M., Reynolds, J.C., eds.: Algebraic methods in semantics. Cambridge University Press (1985) 169–236
11. Ariola, Z.M.: Relating graph and term rewriting via Böhm models. Applicable Algebra in Engineering, Communication and Computing **7** (1996) 401–426
12. Kennaway, R., van Oostrom, V., de Vries, F.J.: Meaningless terms in rewriting. The Journal of Functional and Logic Programming **1** (1999)
13. Baader, F., Nipkow, T.: Term Rewriting and All That. Cambridge University Press (1998)
14. Stoltenberg-Hansen, V., Lindström, I., Griffor, E.R.: Mathematical Theory of Domains. Cambridge University Press (1994)
15. Klop, J.W., Middeldorp, A.: Sequentiality in orthogonal term rewriting systems. Journal of Symbolic Computation **12** (1991) 161–195
16. Huet, G., Lévy, J.J.: Computations in orthogonal rewriting systems. In Lassez, J.L., Plotkin, G., eds.: Computational Logic. MIT Press (1991) 395–443