Bohm-Like Trees for Rewriting

Ketema, J.

2006

Böhm-Like Trees for Rewriting

Jeroen Ketema

IPA dissertation series 2006-07

Typeset with LaTeX2$_\varepsilon$
Cover design by Dries Verbruggen, unfold.be
Printed by PrintPartners Ipskamp, Enschede, The Netherlands

The work in this thesis has been carried out under the auspices of the research school IPA (Institute for Programming research and Algorithmics).

VRIJE UNIVERSITEIT

Böhm-Like Trees for Rewriting

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad Doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. T. Sminia,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de faculteit der Exacte Wetenschappen
op maandag 20 maart 2006 om 15.45 uur
in de aula van de universiteit,
De Boelelaan 1105

door

Jeroen Ketema

geboren te Schagen

promotor:	prof.dr. J.W. Klop
copromotoren:	dr. R.C. de Vrijer
	dr. F. van Raamsdonk

Thanks everyone!

Contents

1

Introduction

Don't Panic. It's the first helpful or intelligible thing anybody's said to me all day.

— Douglas Adams
The Hitchhiker's Guide to the Galaxy (1979)

In this dissertation, we study *denotational semantics* of rewrite systems based on terms. That is, we study maps from terms to some mathematical domain. More specifically, we define classes of denotational semantics that satisfy certain interesting properties.

Given that rewrite systems based on terms can be considered to represent programming languages, where terms are programs, it follows that our approach to denotational semantics differs from the usual approach. In the usual approach, one picks a particular programming language and one defines denotational semantics for this language. Subsequently, one studies the properties of the defined denotational semantics. Examples of the usual approach can be found in the work by De Bakker and De Vink [BV96], by Plotkin [Plo77], and by Winskel [Win93], who all define denotational semantics for 'toy' languages. It can also be found in the work by Jacobs and Poll [JP03] who define denotational semantics for the Java programming language.

Before we can pursue the approach to denotational semantics as taken in this dissertation, i.e., before we can define classes of denotational semantics for rewrite systems, four questions need to be answered:

– Which notion of rewrite systems is employed?
– Which mathematical domain is chosen?
– Which objects from the domain are assigned to which terms?
– What are interesting properties of denotational semantics?

The above questions are answered in Section 1.1. In Section 1.2, we briefly survey the results that have previously been obtained with respect to the approach to denotational semantics as taken in this dissertation. Thereafter, in Section 1.3, we discuss the contributions of this dissertation. Finally, in Section 1.4, we outline the contents of the remaining chapters.

1.1 Domain of Discourse

We next discuss in turn the four questions mentioned above. We start with the employed rewrite systems. Following this, we discuss the mathematical domain

and the way in which we roughly want to assign objects to terms. Finally, we discuss the properties we want our classes of denotational semantics to satisfy.

1.1.1 Rewrite Systems

With respect to rewrite systems based on terms, there are essentially two options: *Term Rewriting Systems (TRSs)* and *Higher-Order Rewriting Systems (HORSs)*. Given the options, we prefer to use TRSs in most of this dissertation. The reason is pragmatic. Thus far, very little theory has been developed regarding the approach to denotational semantics as taken in this dissertation (see Section 1.2). Hence, since TRSs are simpler than HORSs, as they do not allow variable binding, it seems most sensible to first develop more theory regarding TRSs and to consider HORSs only thereafter.

The relative preference of TRSs over HORSs is reflected by the contents of this dissertation: In the main part of this dissertation, theory is developed for TRSs. Only in the last chapter do we consider HORSs. More specifically, in the last chapter Higher-Order Rewrite Systems (HRSs) are considered. These systems are a special flavour of HORSs developed by Nipkow [Nip91].

Besides TRSs and HORSs, there is a third option with respect to the employed rewrite systems: *Abstract Reduction Systems (ARSs)*. However, as ARSs are not based on terms, they have little to do with programming languages. Hence, we disregard them here.

There is also a more pragmatic reason to disregard ARSs: Two of the properties of denotational semantics we are interested in, congruence and sequentiality (see Section 1.1.3), explicitly make use of terms. Hence, to be able to scrutinize these properties in the context of ARSs requires a generalisation of the properties that is independent of terms. Since the development of such independent definitions is notoriously difficult, employing terms seems more sensible in view of obtaining actual results regarding classes of denotational semantics.

1.1.2 Mathematical Domain and Denotational Semantics

Regarding the mathematical domain whose objects we assign to terms, we prefer to take a minimalistic stance. That is, we prefer to choose a mathematical domain we have already available, thereby avoiding the introduction of any new domains.

Given that we consider rewrite systems based on terms, our minimalistic stance implies that there is one obvious choice for the mathematical domain: the set of terms. This immediately suggests a possible way of assigning objects to terms, taken into account the reduction rules of the considered (confluent) rewrite system: Assign to each term its normal form with respect to the considered rewrite system. Unfortunately, this way of assigning objects to terms has one very important shortcoming: not every term needs to have a normal form.

The above shortcoming may manifest itself in one of two ways: either terms grow unbounded or they have non-erasable subterms whose reducts always reduce to a redex. The first way in which the shortcoming may manifest itself can be observed when considering the following rewrite rule:

$$c \to f(c)\,.$$

The rewrite rule yields the reduction:

$$c \to f(c) \to f(f(c)) \to f(f(f(c))) \to \cdots\,,$$

where the dots mean that the reduction continues *ad infinitum*. Obviously, as c occurs as a subterm of each of the terms in the reduction, we have that none of the terms is a normal form.

The second way in which the shortcoming may manifest itself can be observed when considering the following two rewrite rules:

$$a \to b$$
$$b \to a$$

These rewrite rules yield the reduction:

$$a \to b \to a \to b \to \cdots\,.$$

That is, both a and b reduce to themselves in two steps. Since these are the only possible reductions, neither a nor b has a normal form.

In the case of unbounded growth, we can think of the result of the reduction as a term consisting of an infinite nesting of f symbols. Obviously, we can deal with this manifestation of not having a normal form by allowing terms that are infinitely large. That is, by allowing *infinite terms* or *infinite trees* (that are finitely branching). The intuition is that terms that grow unbounded represent programs that return an infinite amount of data in an infinite amount of time. Hence, they should be represented in the chosen mathematical domain. Of course, by allowing terms to be infinite, the usual inductive definition of terms no longer suffices.

In the case of the subterm whose reducts always reduce to a redex, we can deal with the lack of having a normal form by replacing the 'offending' subterm by $\bot$, where $\bot$ is some fresh symbol. With respect to the example, this implies that both a and b must be replaced by $\bot$. The intuition is that terms whose reducts always reduce to a redex represent programs that do not return any result. Hence, we cannot distinguish between such programs from the outside and we may assign identical objects to these programs.

Given the above, we choose as our mathematical domain the set of infinite terms over the assumed signature extended with a fresh symbol $\bot$. Hence, a denotational semantics in this dissertation is understood to be a map from a set of terms to a set of infinite terms. The map assigns to each term its 'normal form', which may be infinite and which may only exist in case certain subterms are replaced by $\bot$. Of course, factually, this only defines a single denotational semantics. We obtain different denotational semantics by also replacing by $\bot$ certain selected subterms whose reducts do not always reduce to a redex.

The denotational semantics presented above derives from a particular approach to denotational semantics that occurs in the $\lambda\beta$-calculus: The so-called *Böhm trees* [Lév78, Bar84], *Lévy-Longo trees* [Lév75, Lon83], and *Berarducci trees* [Ber96] each

define a denotational semantics in the above sense. The trees differ in the particular subterms replaced by $\bot$.

Collectively we call the trees of the $\lambda\beta$-calculus the *Böhm-like trees*, named after the Böhm trees, which are most well-known. This also explains the title of this dissertation: Since the denotational semantics we consider derives from the Böhm-like trees of the $\lambda\beta$-calculus, we prefer to call our denotational semantics *Böhm-like trees for TRSs* and *Böhm-like trees for HORSs*. Hence, in general, *Böhm-like trees for rewriting*.

1.1.3 Properties

Since our approach to denotational semantics originates in the Böhm-like tree approach of the $\lambda\beta$-calculus, it is natural to derive the properties we want our denotational semantics to satisfy from the properties satisfied by (some of) the Böhm-like trees of the $\lambda\beta$-calculus. This is exactly what we do.

Denoting the set of terms and infinite terms respectively by $\mathcal{T}er$ and $\mathcal{T}er^\infty$ and representing a denotational semantics in our sense by $\mathrm{BLT} : \mathcal{T}er \to \mathcal{T}er^\infty$, we briefly discuss the properties derived from the Böhm-like trees of the $\lambda\beta$-calculus that are studied in this dissertation.

Model. The $\lambda\beta$-calculus, and also TRSs and HORSs, are essentially (higher-order) equational logics in which the equations have been directed. Correspondingly, we have the property of the denotational semantics being a *model* of the rewrite system under consideration. By definition of a model, this actually implies the existence of two properties:

Preservation. This property states that the object assigned to a term is preserved under rewriting:

$$s \to^* t \text{ implies } \mathrm{BLT}(s) = \mathrm{BLT}(t)\,,$$

where $s, t \in \mathcal{T}er$ and where $\to^*$ denotes the transitive-reflexive closure of the rewrite relation.

Congruence. This property states that equality of objects assigned to terms is preserved under contexts:

$$\mathrm{BLT}(s) = \mathrm{BLT}(t) \text{ implies } \mathrm{BLT}(C[s]) = \mathrm{BLT}(C[t])\,,$$

where $s, t \in \mathcal{T}er$ and where $C[\Box]$ is a context. Remark that this property requires the syntactic notion of a context, as hinted at in Section 1.1.1 when ruling out ARSs.

Technical Properties. Proving that a certain denotational semantics satisfy congruence often involves the study of certain properties of a technical nature. A prerequisite for the formulation of these properties is the existence of an order on both terms and infinite terms, where the assumed signature is extended with a fresh symbol $\bot$. In both instances $\bot$ must be the least element with respect to the order and, denoting the order by $\preccurlyeq$, it must also hold that $f(s_1, \ldots, s_n) \preccurlyeq f(t_1, \ldots, t_n)$ if and only if $s_i \preccurlyeq t_i$ for all $1 \leqslant i \leqslant n$.

The technical properties are as follows:

Monotonicity. This property states that the denotational semantics preserves the order on terms:

$$s \preccurlyeq t \text{ implies } \mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)\,,$$

where $s, t \in \mathcal{T}er$.

Continuity. This property states that the denotational semantics is continuous:

$$\mathrm{BLT}(s) = \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq s\}\,,$$

where $s \in \mathcal{T}er$, where $\bigsqcup$ denotes the least upper bound, and where it is of course required that the least upper bound exists. The property derives from the order theoretic notion of continuity, as explained, e.g., by Amadio and Curien [AC98].

Syntactic Continuity. Assuming that the set of terms can be embedded in the set of infinite terms by means of a map ι, this property states that contexts are continuous:

$$\mathrm{BLT}(C[s]) = \bigsqcup\{\mathrm{BLT}(C[t]) \mid \iota(t) \preccurlyeq \mathrm{BLT}(s)\}\,,$$

where $s \in \mathcal{T}er$, where $C[\Box]$ is a context, and where it is again required that the least upper bound exists. The property is easily shown to imply congruence. Contrary, however, it is possible to define denotational semantics that satisfy congruence but not syntactic continuity. As in the case of continuity, this property derives from the order theoretic notion of continuity, as also explained by, e.g., Amadio and Curien [AC98].

Sequentiality. Some rewrite systems are inherently *non-concurrent*. Hence, they are sequential. The sequentiality property expresses this fact. The actual definition of the property is quite complex. Therefore, it is not described here any further. A thorough explanation can be found in Chapter 8. Note, however, that the property depends on the presence of terms, as hinted at in Section 1.1.1 when ruling out ARSs.

1.2 Previous Work

The previous work concerning Böhm-like trees for rewriting, i.e., concerning the approach to denotational semantics as outlined above, can roughly be divided into two categories depending on the way in which the trees are actually defined. We deal with each of these in turn.

Direct Approximants. To define Böhm-like trees by means of *direct approximants* requires the definition of a map that assigns to each term a 'partial' term. Here, a partial term consists of that part of a term that also occurs in its 'normal form'. The Böhm-like tree of a term is now defined as the set of partial terms assigned to the reducts of the term under consideration. Different Böhm-like trees are obtained by varying the definition of the map that assigns partial terms to terms.

Regarding the direct approximant definition, the following results have previously been obtained:

Preservation. Boudol [Bou85], Blom [Blo01], and Ariola and Blom [AB02] each have defined Böhm-like trees satisfying preservation under rewriting. In the case of Boudol's work, this concerns Böhm-like trees for TRSs. In the case of both Blom's work and the work by Ariola and Blom, this concerns Böhm-like trees for ARSs.

Monotonicity and Continuity. The Böhm-like trees defined by Boudol [Bou85] do not only satisfy preservation under rewriting, but also monotonicity and continuity.

Congruence and Syntactic Continuity. Ariola [Ari96] has defined a *concrete* Böhm-like tree for TRSs which satisfies both congruence of Böhm-like tree equality and syntactic continuity. Blom [Blo01] has done the same for Combinatory Reduction Systems (CRSs), a particular kind of HORSs developed by Klop [Klo80].

Infinitary Rewriting. To define Böhm-like trees by means of *infinitary rewriting* requires the extension of the considered rewrite system with both infinite terms and infinite reductions. Moreover, it requires the addition of a number of rewrite rules ensuring that each term has a normal form. As such, the object assigned to a term is defined as the normal form of the term with respect to the extended system. Different Böhm-like trees are obtained by varying the rules which ensure that each term has a normal form.

Regarding the infinitary rewriting definition, the following result has previously been obtained:

Preservation and Congruence. Kennaway, Van Oostrom, and De Vries [KOV99] have defined Böhm-like trees that satisfy both preservation under rewriting and congruence.

1.3 Contributions

The two most important contributions of this dissertation are as follows:

- Employing direct approximants, a number of Böhm-like trees for TRSs are defined. Incrementally, the trees satisfy preservation under rewriting, monotonicity and continuity, congruence and syntactic continuity, and sequentiality. The definitions are such that most results previously obtained are generalised.
- A comparison is made between Böhm-like trees defined by means of direct approximants and Böhm-like trees defined by means of infinitary rewriting. It is shown that all Böhm-like trees definable by means of infinitary rewriting can also be defined by means of direct approximants.

In addition to the above, the following contributions are also made:

- Three well-known representations of infinite terms are compared with the help of coalgebras. It is shown that the three definitions are equivalent. In addition, it is shown that the same holds for three particular maps defined on infinite terms.
- Known definitions and properties of the three Böhm-like trees of the $\lambda\beta$-calculus are surveyed. In addition, a number of Böhm-like trees for PCF and the $\lambda\beta\eta$-calculus are presented.

– The Böhm-like trees for TRSs, as defined by means of direct approximants, are extended to HRSs as far as preservation of rewriting, monotonicity, and continuity are concerned.

1.4 Outline

The dependencies between the chapters that make up the remainder of this dissertation are depicted in Figure 1.1. The dotted arrows in the figure denote a dependency on properties satisfied by the Böhm-like trees of the $\lambda\beta$-calculus. The arrows labelled with a section number denote a dependency on a particular section.

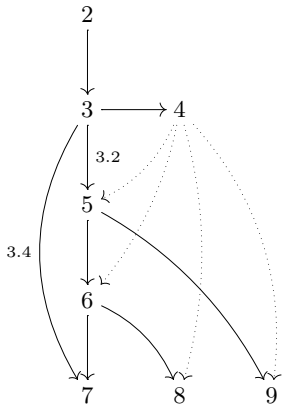

Figure 1.1. Dependencies between chapters

The contents of each of the chapters is roughly as follows:

Chapter 2. The notions and notation employed in the other chapters of this dissertation are established.

Chapter 3. A general coalgebraic definition of infinite terms is given and it is shown for three well-known representations of infinite terms that they define final coalgebras.

Chapter 4. The known Böhm-like trees of the $\lambda\beta$-calculus and their properties are surveyed. In addition, a number of Böhm-like trees for PCF and the $\lambda\beta\eta$-calculus are presented.

Chapter 5. A general definition of Böhm-like trees for TRSs is presented of which it is shown that it satisfies preservation under rewriting. In addition, Böhm-like trees are defined that satisfy monotonicity and continuity.

Chapter 6. Böhm-like trees are defined which satisfy both congruence and syntactic continuity.

Chapter 7. The Böhm-like trees defined in Chapters 5 and 6 are compared with the Böhm-like trees for TRSs defined by means of infinitary rewriting.

Chapter 8. Böhm-like trees are defined that are sequential. Sequentiality is also defined in this chapter, as it is not used anywhere else in this dissertation.

Chapter 9. The general definition of Böhm-like trees for TRSs, as presented in Chapter 5, is extended to HRSs. The same is done for Böhm-like trees that satisfy monotonicity and continuity.

2

Preliminaries

This is incredible! I heard rumors about it before!

— Douglas Adams
The Hitchhiker's Guide to the Galaxy (1979)

We establish the notions and notation employed throughout this dissertation. It is explicitly not the aim of this chapter to explain in depth each notion introduced. If the reader wishes to gain more knowledge of a particular subject, he or she is advised to consult the cited works.

2.1 Sets, Strings, and Topologies

In this section, we introduce the preliminaries regarding sets, strings, and topologies. With respect to sets Halmos' classic textbook [Hal60] suffices for this dissertation. Of course, any basic text on axiomatic set theory, such as Chapter 9 of Shoenfield's book [Sho67], will also do. With respect to strings any textbook on formal language theory may be consulted, e.g., those by Linz [Lin96] and by Davis, Sigal, and Weyuker [DSW94]. Finally, with respect to topologies any textbook on general topology, such as the one by Kelley [Kel75], suffices.

2.1.1 Sets

The following table summarises the notation employed for particular sets:

Notation	Meaning
$\emptyset$	the *empty set*
$\wp(X)$	the *power set* of X
$X \times Y$	the *Cartesian product* of X and Y
$X - Y$	the *difference* of X and Y
$\mathbb{N}$	the set of *natural numbers* including 0
$\mathbb{R}$	the set of *real numbers*
$\mathbb{R}^+$	the set of *positive real numbers*

In addition to the above, we use $\#X$ to denote the cardinality of a set X. We call a set *finite* if it has finite cardinality. Otherwise, we call it *infinite*.

By α, β, γ, and κ we denote arbitrary ordinals. A *limit ordinal* is any ordinal α such that $\alpha \neq 0$ and such that there does not exist an ordinal β with $\alpha = \beta + 1$. We denote the first limit ordinal by ω.

Remember that each ordinal α denotes the set of all ordinals smaller than α. In accordance, we sometimes write $\beta \in \alpha$ for $\beta < \alpha$ and we also employ α as the (co)domain of certain maps.

Given a map $f : X \to Y$ and a set $Z \subseteq Y$, the *inverse image* of Z under f is defined as:

$$f^{-1}[Z] = \{x \in X \mid f(x) \in Z\}.$$

A *(binary) relation* over sets X and Y is a set $R \subseteq X \times Y$. If $(x, y) \in R$, then we write $x\ R\ y$. The *converse* of a relation R over X and Y is the relation over Y and X denoted R^{-1}. We have $x\ R^{-1}\ y$ if and only if $y\ R\ x$.

A *sequence* over a set X is a map from some ordinal α to X. We denote a sequence by $(x_\kappa)_{\kappa<\alpha}$. Moreover, we denote $(x_\kappa)_{\kappa<\alpha}(\gamma)$ by x_γ for all $\gamma < \alpha$. A sequence $(x_\kappa)_{\kappa<\alpha}$ is called *finite* if $\alpha < \omega$.

A *subsequence* of $(x_\kappa)_{\kappa<\alpha}$ is a sequence $(y_\kappa)_{\kappa<\beta}$ such that there exists a map $\iota : \beta \to \alpha$ with $y_\gamma = x_{\iota(\gamma)}$ for all $\gamma < \beta$, and $\iota(\gamma_1) < \iota(\gamma_2)$ if and only if $\gamma_1 < \gamma_2$. A subsequence is called an *initial sequence*, if $\iota(\gamma) = \gamma$ for all $\gamma < \beta$.

2.1.2 Strings

A *string* over a set X is a finite sequence over X. In this case X is called an *alphabet*. The set of all strings over X is denoted X^*. The *empty string*, i.e., the sequence whose domain is 0, is denoted ϵ. Usually, we write $x_0x_1 \cdots x_{n-1}$ for a string $(x_i)_{i<n}$. The *length* of a string $s = (x_i)_{i<n}$, denoted $|s|$, is n.

Given $s = (x_i)_{i<m}$ and $t = (y_i)_{i<n}$, we define the concatenation of s and t, denoted $s \cdot t$, as the sequence $(z_i)_{i<m+n}$ such that $z_j = x_j$ for all $j < m$ and $z_j = y_{j-m}$ for all $m \leqslant j < m + n$. The concatenation of an element $x \in X$ and a string s is defined as the concatenation of the string $(x_i)_{i<1}$, where $x_0 = x$, and s. The empty string is the neutral element with respect to concatenation. That is, $\epsilon \cdot s = s = s \cdot \epsilon$.

For each $s \in X^*$ and $n \in \mathbb{N}$ we inductively define the *exponentiation* of s to n, denoted s^n, as:

$$s^n = \begin{cases} \epsilon & \text{if } n = 0 \\ s \cdot s^{n-1} & \text{if } n > 0 \end{cases}$$

Strings have an associated *prefix order*: The string s is a *prefix* of a string t, denoted $s \leqslant t$, if there exists a string u such that $s \cdot u = t$. If $s \leqslant t$, we also write $t \geqslant s$. Moreover, if $s \leqslant t$ and $s \neq t$, then we write $s < t$ and $t > s$. We say that s and t are *parallel*, denoted $s \parallel t$, if neither $s \leqslant t$ nor $t \leqslant s$.

Given a string s and a natural number n, we define the *truncation* to n of s, denoted $s[n]$, as:

$$s[n] = \begin{cases} \epsilon & \text{if } n = 0 \text{ or } s = \epsilon \\ x \cdot (s'[n-1]) & \text{if } n > 0 \text{ and } s = x \cdot s' \end{cases}$$

Lifting the above definition to sets of strings, we define the *truncation* to n of set S of strings, denoted $S[n]$, as:

$$S[n] = \{s[n] \mid s \in S\}.$$

Finally, given a set of strings $S \subseteq X^*$ and an element $x \in X$, define the set of *suffixes* of S with respect to x, denoted $S|_x$, as:

$$S|_x = \{s \in X^* \mid x \cdot s \in S\}.$$

2.1.3 Topologies

A *topology* for a set X is a set $\mathcal{T} \subseteq \wp(X)$ such that:

1. $\emptyset, X \in \mathcal{T}$,
2. if $\mathcal{O} \subseteq \mathcal{T}$ such that $\mathcal{O}$ is finite, then $\bigcap \mathcal{O} \in \mathcal{T}$, and
3. if $\mathcal{O} \subseteq \mathcal{T}$, then $\bigcup \mathcal{O} \in \mathcal{T}$.

The pair $(X, \mathcal{T})$ is called a *topological space*. The elements of $\mathcal{T}$ are called *open sets*. A set $C \subseteq X$ is called *closed* if $X - C$ is open. The set $\wp_{\mathrm{nc}}(X)$ is the set of all *non-empty* closed subsets of X. If $Y \subseteq X$, then the *closure* of Y, denoted $\mathrm{Cl}(Y)$, is defined as:

$$\mathrm{Cl}(Y) = \bigcap \{C \mid C \text{ closed and } Y \subseteq C\}$$

It is a well-known fact that Y is closed if and only if $Y = \mathrm{Cl}(Y)$.

Given a topological space $(X, \mathcal{T})$, we call a set $\mathcal{B} \subseteq \wp(X)$ a *basis* of $\mathcal{T}$, if:

1. $\mathcal{B} \subseteq \mathcal{T}$, and
2. for every $O \in \mathcal{T}$ there exists a $\mathcal{B}' \subseteq \mathcal{B}$ such that $O = \bigcup_{B' \in \mathcal{B}'} B'$.

For each basis $\mathcal{B}$, if $B_1, B_2 \in \mathcal{B}$ and $x \in B_1 \cap B_2$, then there exists a $B \in \mathcal{B}$ such that $x \in B \subseteq B_1 \cap B_2$.

A typical example of a basis is the one of the *order topology* for an ordinal α. The basis is the set of all open intervals. That is, it consists of the sets:

$$\begin{aligned}(\beta, \gamma) &= \{\kappa \in \alpha \mid \beta < \kappa < \gamma\} \\ (\beta, \rightarrow) &= \{\kappa \in \alpha \mid \kappa > \beta\} \\ (\leftarrow, \beta) &= \{\kappa \in \alpha \mid \kappa < \beta\} \\ (\leftarrow, \rightarrow) &= \alpha\end{aligned}$$

The topological space consisting of an ordinal α and the order topology for α is called the *ordinal space* of α.

If $(X, \mathcal{T}_X)$ and $(Y, \mathcal{T}_Y)$ are topological spaces, then a map $f : X \to Y$ is called *continuous* if for all $O \in \mathcal{T}_Y$ it holds that $f^{-1}[O] \in \mathcal{T}_X$. Moreover, f is called a *homeomorphism* if f is an isomorphism and if both f and f^{-1} are continuous.

Given a set X, a map $d : X \times X \to \mathbb{R}$ is called a *metric* or *distance function* on X, if for all $x, y, z \in X$ it holds that:

1. $d(x, y) \geqslant 0$,
2. $d(x, y) = 0$ if and only if $x = y$,
3. $d(x, y) = d(y, x)$, and
4. $d(x, y) + d(y, z) \geqslant d(x, z)$ (*triangle inequality*).

A *metric space* is a pair (X, d) such that d is a metric or distance function on X. In the remainder of this dissertation we are mostly interested in metric spaces.

Given a set X, the *discrete metric* d_{d} is defined for all $x, y \in X$ as:

$$d_{\mathrm{d}}(x,y) = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{if } x \neq y \end{cases}$$

A proof that d_{d} is actually a metric can be found in any text on metric topology.

Given a metric space (X,d), we define for all $x \in X$ and $r \in \mathbb{R}^+$ the *open ball* of x with *radius* r:

$$B(x,r) = \{y \in X \mid d(x,y) < r\}\,.$$

The set $B(x,r)$ is sometimes also called the *open sphere* of x with radius r.

It is a well-known fact that each metric space (X,d) defines a topological space $(X,\mathcal{T}_d)$, where $O \in \mathcal{T}_d$ if and only if there exists for every $x \in O$ a real number $r \in \mathbb{R}^+$ such that $B(x,r) \subseteq O$. Moreover, it is well-known that the set:

$$\{B(x,r) \mid r \in \mathbb{R}^+ \text{ and } x \in X\}$$

is a basis of $\mathcal{T}_d$.

Given a metric space (X,d) such that there exists an $r \in \mathbb{R}^+$ with $d(x,y) < r$ for all $x, y \in X$, the *Hausdorff metric* d_{H} over the set $\wp_{\mathrm{nc}}(X)$ is defined as:

$$d_{\mathrm{H}}(Y,Z) = \bigsqcap\{r \mid Y \subseteq V_r(Z) \text{ and } Z \subseteq V_r(Y)\}\,,$$

where $\bigsqcap$ denotes the greatest lower bound (see also Section 2.3) and where V_r is defined as:

$$V_r(Y) = \{x \in X \mid d(x,Y) < r\}$$

with $d(x,Y)$ as follows:

$$d(x,Y) = \bigsqcap\{d(x,y) \mid y \in Y\}\,.$$

A proof that the Hausdorff metric actually is a metric can, e.g., be found in Kelley's book [Kel75] and also in the book by De Bakker and De Vink [BV96].

We next define convergence, limits, and continuity. The definitions slightly generalise the usual ones by allowing sequences of arbitrary ordinal length. Although we shall not explore this matter any further here, the definitions given below, and the usual ones too, are instances of even more general definitions. These definitions are based the notion of a *net*. More on this subject can be found in Kelley's book [Kel75].

Definition 2.1.1. *Let (X,d) be a metric space, α an ordinal, and $(x_\kappa)_{\kappa<\alpha}$ a sequence over X. The sequence is* convergent*, if there exists an element $x \in X$ such that for every $\epsilon \in \mathbb{R}^+$ an ordinal $\beta_\epsilon < \alpha$ exists with $d(x,x_\gamma) < \epsilon$ for all $\beta_\epsilon < \gamma < \alpha$. The element x, also denoted* $\lim (x_\kappa)_{\kappa<\alpha}$*, is called a* limit *of the sequence.*

Alternatively, but equivalently, x is called a limit if there exists for every $\epsilon \in \mathbb{R}^+$ an ordinal $\beta_\epsilon < \alpha$ such that:

$$\{x_\gamma \mid \beta_\epsilon < \gamma < \alpha\} \subseteq B(x,\epsilon)\,.$$

It is a well-known fact that a sequence can have at most one limit. Hence, we can speak of *the* limit of a sequence, if the sequence has a limit.

Definition 2.1.2. *Let (X, d) be a metric space and α an ordinal. A sequence $(x_\kappa)_{\kappa<\alpha}$ of elements of X is* continuous*, if for every limit ordinal $\beta < \alpha$ it holds that x_β is the limit of the initial sequence $(x_\kappa)_{\kappa<\beta}$ of $(x_\kappa)_{\kappa<\alpha}$.*

Convergence of a continuous sequence of length α can be expressed by means of continuity: There must exist a continuous sequence of length $\alpha+1$ such that the original sequence is an initial sequence.

That the definition of continuity is correct, given the definition of continuous functions, is the contents of the following proposition:

Proposition 2.1.3. *Let (X, d) be a metric space. A sequence $(x_\kappa)_{\kappa<\alpha}$ is continuous if and only if the map f from the ordinal space of α to $(X, \mathcal{T}_d)$ defined for all $\beta < \alpha$ by $f(\beta) = x_\beta$ is continuous.*

Proof. Let $(x_\kappa)_{\kappa<\alpha}$ be a continuous sequence and suppose $B(x, r)$ is an open ball of (X, d). By definition of the sequence, $f^{-1}[B(x, r)]$ consists of a union of open intervals. Hence, as the set of open intervals is a basis of the order topology on α, we have that $f^{-1}[B(x, r)]$ is open. Since the set of open balls forms a basis of $\mathcal{T}_d$, it follows that f is continuous.

Now suppose f is continuous and let $\gamma < \alpha$ be a limit ordinal. For an arbitrary $\epsilon \in \mathbb{R}^+$ consider the open ball $B(f(\gamma), \epsilon)$. By definition of continuity and the order topology, we have that $f^{-1}[B(f(\gamma), \epsilon)]$ consists of a number of open intervals. As $\gamma \in f^{-1}[B(f(\gamma), \epsilon)]$, there must exist an open interval $(\beta_\epsilon, \gamma) \subseteq f^{-1}[B(f(\gamma), \epsilon)]$. Hence, $\{x_\kappa \mid \beta_\epsilon < \kappa < \gamma\} \subseteq B(f(\gamma), \epsilon)$. As ϵ was arbitrary, we have that $f(\gamma) = x_\gamma$ is the limit of the initial sequence $(x_\kappa)_{\kappa<\gamma}$. □

Besides continuous sequences, we also need Cauchy sequences. Contrary to the definition of continuous sequences, we do not generalise the definition of Cauchy sequences to arbitrary ordinals. This is not required in the remaining chapters of this dissertation. For a generalised definition based on nets, see again Kelley's book [Kel75].

Definition 2.1.4. *Let (X, d) be a metric space. A sequence $(x_\kappa)_{\kappa<\omega}$ of elements of X is called a* Cauchy sequence *of X, if there exists for every $\epsilon \in \mathbb{R}^+$ an ordinal $\beta_\epsilon < \omega$ such that $d(x_{\gamma_1}, x_{\gamma_2}) < \epsilon$ for all $\beta_\epsilon < \gamma_1, \gamma_2 < \omega$. The metric space is called* complete *if every Cauchy sequence has a limit.*

Each metric space (X, d_{d}), with d_{d} the discrete metric, is complete.

Given that a map $f : X \to Y$ between metric spaces (X, d_X) and (Y, d_Y) is called *isometric* if for all $x_1, x_2 \in X$ it holds that $d_X(x_1, x_2) = d_Y(f(x_1), f(x_2))$, the following theorem is well-known:

Theorem 2.1.5. *Let (X, d) be a metric space. There exists a complete metric space $(X^\star, d^\star)$ and an isometric map $e : (X, d) \to (X^\star, d^\star)$ such that $(X^\star, d^\star)$ is unique up to isometric homeomorphisms.*

The set $X^\star$ can be defined as the set of all equivalence classes of Cauchy sequences over X, where two Cauchy sequences $(x_\kappa)_{\kappa<\omega}$ and $(y_\kappa)_{\kappa<\omega}$ are equal if

and only if $\lim_{\kappa\to\omega}\ d(x_\kappa, y_\kappa) = 0$. The equivalence class with $(x_\kappa)_{\kappa<\omega}$ as a representative is denoted $[\![(x_\kappa)_{\kappa<\omega}]\!]$. As such, $d^\star([\![(x_\kappa)_{\kappa<\omega}]\!], [\![(y_\kappa)_{\kappa<\omega}]\!])$ is defined as $\lim_{\kappa\to\omega}\ d(x_\kappa, y_\kappa)$. The value of $d^\star([\![(x_\kappa)_{\kappa<\omega}]\!], [\![(y_\kappa)_{\kappa<\omega}]\!])$ does not depend on the chosen representatives.

For a proof of the above, see, e.g., Theorem 6.27 in Kelley's book [Kel75]. The metric space $(X^\star, d^\star)$ is called the *metric completion* of (X, d).

2.2 Term Rewriting

In this section, we give the relevant definitions regarding term rewriting. The section consists of two parts: The first part deals with terms. The second part deals with rewriting. Relevant books on term rewriting are Baader and Nipkow's textbook [BN98] and the book by Terese [Ter03].

2.2.1 Terms

A *signature* Σ is a set of elements called *function symbols*. Each function symbol is assigned a natural number, called the *arity* of the symbol. Function symbols of arity zero are called *nullary* function symbols. Function symbols of arity one and two are called respectively *unary* and *binary* function symbols.

If $f \in \Sigma$, we denote the arity of f by $ar(f)$. We also write:

$$\Sigma_n = \{f \in \Sigma \mid ar(f) = n\}\,.$$

By f, g, h, ... we denote arbitrary function symbols and by a, b, c, ... arbitrary nullary function symbols.

The set $\mathcal{T}er(\Sigma, V)$ of *terms* over a signature Σ and a countably infinite set of *variables* V is inductively defined as follows:

- $x \in \mathcal{T}er(\Sigma, V)$, if $x \in V$, and
- $f(s_1, \ldots, s_n) \in \mathcal{T}er(\Sigma, V)$, if $n \in \mathbb{N}$, $f \in \Sigma_n$, and $s_1, \ldots, s_n \in \mathcal{T}er(\Sigma, V)$.

By x, y, z, ... we denote arbitrary variables and by s, t, ... arbitrary terms. We call $s \in \mathcal{T}er(\Sigma, V)$ *linear* if each variable occurs at most once in s.

We next define a number of maps on $\mathcal{T}er(\Sigma, V)$. In the definition the set $\mathbb{N}^*$, i.e., the set of all the strings over the natural numbers, is employed.

Definition 2.2.1. *Let $s, t \in \mathcal{T}er(\Sigma, V)$ and let $p \in \mathbb{N}^*$.*

- *The* root symbol *of s, denoted $root(s)$, is defined as:*

$$root(s) = \begin{cases} x & \text{if } s = x \text{ and } x \in V \\ f & \text{if } s = f(s_1, \ldots, s_n) \end{cases}$$

- *The* set of variables *of s, denoted $\mathcal{V}ar(s)$, is inductively defined as:*

$$\mathcal{V}ar(s) = \begin{cases} \{x\} & \text{if } s = x \text{ and } x \in V \\ \bigcup_{i=1}^{n} \mathcal{V}ar(s_i) & \text{if } s = f(s_1, \ldots, s_n) \end{cases}$$

– *The set of* positions *of s, denoted $\mathcal{P}os(s)$, is inductively defined as:*

$$\mathcal{P}os(s) = \begin{cases} \{\epsilon\} & \text{if } s \in V \\ \{\epsilon\} \cup \bigcup_{i=1}^{n} \{i \cdot p \mid p \in \mathcal{P}os(s_i)\} & \text{if } s = f(s_1, \ldots, s_n) \end{cases}$$

– *If $p \in \mathcal{P}os(s)$, then the* subterm *of s at position p, denoted $s|_p$, is inductively defined as:*

$$s|_p = \begin{cases} s & \text{if } p = \epsilon \\ s_i|_q & \text{if } p = i \cdot q \text{ and } s = f(s_1, \ldots, s_n) \end{cases}$$

where $|p|$ is called the depth *of $s|_p$ in s.*

– *If $p \in \mathcal{P}os(s)$, then the* replacement *by t of the subterm at position p in s, denoted $s[t]_p$, is inductively defined as:*

$$s[t]_p = \begin{cases} t & \text{if } p = \epsilon \\ f(t_1, \ldots, t_n) & \text{if } s = f(s_1, \ldots, s_n),\ p = i \cdot q,\ t_i = s_i[t]_q, \\ & \quad \text{and } t_j = s_j \text{ for all } j \neq i \end{cases}$$

Let Σ be a signature and $\Box$ a fresh nullary function symbol, i.e., nullary function symbol that does not occur in Σ. A *(one-hole) context* is a term over the signature $\Sigma \cup \{\Box\}$ in which $\Box$ occurs precisely once. We denote a context by $C[\Box]$. We call the unique nullary function symbol $\Box$ in $C[\Box]$ the *hole* of the context. Given a term s, we denote by $C[s]$ the context $C[\Box]$ with $\Box$ replaced by s. In terms of subterm replacement: $C[s] = (C[\Box])[s]_p$ if $C[\Box]|_p = \Box$.

A *substitution* is a map $\sigma : V \to \mathcal{T}er(\Sigma, V)$, with $\sigma(x) \neq x$ for only a finite number of variables x. Substitutions are homomorphically extended to terms. That is, the application of σ to a term s, denoted $\sigma(s)$, is defined as:

$$\sigma(s) = \begin{cases} \sigma(x) & \text{if } s = x \text{ and } x \in V \\ f(\sigma(s_1), \ldots, \sigma(s_n)) & \text{if } s = f(s_1, \ldots, s_n) \end{cases}$$

In some cases we employ $x[x_1 := s_1; x_2 := s_2]$ to denote the substitution σ defined as:

$$\sigma(x) = \begin{cases} s_i & \text{if } x = x_i \text{ for } i \in \{1, 2\} \\ x & \text{otherwise} \end{cases}$$

A *renaming* is a substitution σ such that for all $x \in V$ it holds that $\sigma(x) \in V$.

A *unifier* of terms s and t is a substitution σ such that $\sigma(s) = \sigma(t)$ under assumption that $\mathcal{V}ar(s) \cap \mathcal{V}ar(t) = \emptyset$. A unifier σ is said to be more general than a unifier τ, if there exists a substitution ρ such that $\sigma \circ \rho = \tau$. A unifier is a *most general unifier (mgu)* is it is more general than all other unifiers. It is a well-known fact that the existence of a unifier implies the existence of a most general unifier.

We say that s and t *overlap* if one of the following two cases holds:

– a position $p \in \mathcal{P}os(s)$ exists such that $s|_p \notin V$ and $s|_p$ and t unifiable, or
– a position $p \in \mathcal{P}os(t)$ exists such that $t|_p \notin V$ and $t|_p$ and s unifiable.

In the first case we also say that t *overlaps s at $p \in \mathcal{P}os(s)$*. In the second case we also say that s *overlaps t at $p \in \mathcal{P}os(t)$*.

2.2.2 Rewriting

We first introduce the relevant notions from abstract rewriting. Thereafter, the relevant notions from term rewriting are introduced.

An *Abstract Reduction System (ARS)* is a pair $\mathcal{A} = (A, \{\to_i \mid i \in I\})$ with A a set and each $\to_i$, indexed by I, a relation on $A \times A$ called a *rewrite relation*. In later chapters we mostly employ ARSs with $\#I = 1$. We denote these ARSs by $\mathcal{A} = (A, \to)$. Only in one instance are we interested in ARSs with $\#I > 1$ (in Section 5.5). In this case we actually have $\#I = 2$ and we write $\mathcal{A} = (A, \to_i, \to_j)$ for some i and j.

Let $\mathcal{A} = (A, \to)$ be an ARS. The reflexive and transitive-reflexive closures of $\to$ are denoted respectively $\to^=$ and $\to^*$. Moreover, $(\leftarrow) = (\to)^{-1}$, $({}^=\!\leftarrow) = (\to^=)^{-1}$, and $({}^*\!\leftarrow) = (\to^*)^{-1}$. In case $a \to^* b$, we say that a can be *reduced* or *rewritten* to b and that b is a *reduct* of a.

With respect to ARSs, the following properties are important in this dissertation:

Definition 2.2.2. *Let $\mathcal{A} = (A, \to)$ be an ARS. The ARS $\mathcal{A}$ is* subcommutative*, if for every $b_1 \leftarrow a \to b_2$ there exist $a' \in A$ such that $b_1 \to^= a' {}^=\!\leftarrow b_2$ (see Figure 2.1). Moreover, the ARS $\mathcal{A}$ is* confluent*, if for every $b_1 {}^*\!\leftarrow a \to^* b_2$ there exist $a' \in A$ such that $b_1 \to^* a' {}^*\!\leftarrow b_2$ (see Figure 2.2).*

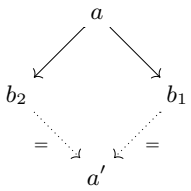

Figure 2.1. Subcommutativity

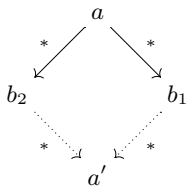

Figure 2.2. Confluence

It is a well-known that an ARS $\mathcal{A} = (A, \to)$ is confluent if it is subcommutative. See, e.g., Lemma 2.7.4 in Baader and Nipkow's book [BN98].

Let $\mathcal{A} = (A, \to)$ be an ARS. A *finite reduction* or *finite rewrite sequence* of length $n \in \mathbb{N}$, denoted $a_0 \to^n a_n$, is a sequence $(a_i)_{i<n+1}$ of elements of A such that $a_i \to a_{i+1}$ for all $i \leqslant n$. The finite reduction is said to *start* in a_0 and *end* in a_n. Obviously, $a \to^* b$ if and only if there exists a finite reduction which starts in a and ends in b. An *infinite reduction* or *infinite rewrite sequence* is a sequence $(a_i)_{i<\omega}$ of elements of A such that $a_i \to a_{i+1}$ for all $i \in \mathbb{N}$. A *reduction* or *rewrite sequence* is either a finite or an infinite reduction.

An ARS $\mathcal{A} = (A, \to)$ is called *terminating* if all reductions are finite. An element $a \in A$ is called a *normal form* of $\mathcal{A}$ if there exists no $b \in A$ such that $a \to b$. By $\mathsf{NF}_{\mathcal{A}}$ we denote the set of all normal forms of $\mathcal{A}$. An ARS $\mathcal{A} = (A, \to)$ is called *normalising* if for each $a \in A$ there exists a reduction starting in a and ending in a normal form of $\mathcal{A}$.

A *rewrite rule* is a pair of terms $(l, r) \in \mathcal{T}er(\Sigma, V) \times \mathcal{T}er(\Sigma, V)$, denoted $l \to r$, such that $root(l) \notin V$ and $\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$. The term l is called the *left-hand side* of the rewrite rule and r is called the *right-hand side*. A rewrite rule is called *left-linear*, if its left-hand side is linear.

Given a rewrite rule $l \to r$ and a substitution σ, we call $\sigma(l)$ an $l \to r$*-redex*. If $s = C[\sigma(l)]$ for some $l \to r$-redex and context $C[\Box]$, with $C[\Box]|_p = \Box$, then we say that an $l \to r$-redex, or simply a redex, *occurs* at position p in s. Moreover, if $q \in \mathcal{P}os(s)$, then q is said to occur in the *redex pattern* of the $l \to r$ redex at position p in s whenever $q \geqslant p$ and not $q \geqslant p \cdot p'$ with $p' \in \mathcal{P}os(l)$ such that $root(l|_{p'}) \in \mathcal{V}ar(l)$.

A pair of terms (s, t), denoted $s \to t$, defines a *rewrite step* over $\mathcal{T}er(\Sigma, V)$, if $s = C[\sigma(l)]$, $t = C[\sigma(r)]$, and if $l \to r$ is a rewrite rule. An $l \to r$-redex is *contracted* in such a step. A set of rewrite steps defines a rewrite relation, denoted $\to$.

A *Term Rewriting System (TRS)* over a signature Σ and a set of rewrite rules R is a pair $\mathcal{R} = (\Sigma, R)$. We call $\mathcal{R}$ *left-linear*, if all its rewrite rules are left-linear. The *rewrite relation* of $\mathcal{R}$, denoted either $\to_{\mathcal{R}}$ or $\to$, is the set of all possible rewrite steps over $\mathcal{T}er(\Sigma, V)$ with respect to the rewrite rules in R. Obviously, $\mathcal{R}' = (\mathcal{T}er(\Sigma, V), \to_{\mathcal{R}})$ is an ARS.

We call a term s *root-active* if every reduct of the term can be reduced to a redex. Moreover, we call s *root-stable* if we *cannot* rewrite s to a redex. We call a subterm $s|_p$ *root-stable* if for all $q \leqslant p$ we have that $s|_q$ is root-stable. Remark that root-stability is undecidable. This follows easily by the undecidability of termination.

Three classes of TRSs are of particular interest to us: (weakly) orthogonal TRSs, constructor TRSs, and approximations.

To define (weakly) orthogonal TRSs, we first define critical pairs. To do so, suppose $l_1 \to r_1$ and $l_2 \to r_2$ are rewrite rules. If, after renaming the variables in the rules such that $\mathcal{V}ar(l_1) \cap \mathcal{V}ar(l_2) = \emptyset$, it holds that l_2 overlaps l_1 at $p \in \mathcal{P}os(l_1)$ with σ as mgu and either the rewrite rules are different or $p \neq \epsilon$, then $\langle \sigma(l_1[r_2]_p), \sigma(r_1) \rangle$ is called a *critical pair*. The critical pair is called *trivial* if $\sigma(l_1[r_2]_p) = \sigma(r_1)$.

We have the following definition:

Definition 2.2.3. *A TRS $\mathcal{R} = (\Sigma, R)$ is called* orthogonal *if all rewrite rules are left-linear and if there are no critical pairs. The TRS is called* weakly orthogonal *if all rewrite rules are left-linear and if all critical pairs are trivial.*

Our interest in (weakly) orthogonal TRSs has to do with the well-known fact these TRSs are confluent.

We next define constructor TRSs:

Definition 2.2.4. *Given a TRS $\mathcal{R} = (\Sigma, R)$, define the following two sets:*

- *the set $\mathcal{D} = \{root(l) \mid l \to r \in R\}$ of* defined symbols*, and*
- *the set $\mathcal{C} = \Sigma - \mathcal{D}$ of* constructors*.*

A rule $f(s_1, \ldots, s_n) \to r$ is called a constructor rule *if $f \in \mathcal{D}$ and $s_1, \ldots, s_n \in \mathcal{T}er(\mathcal{C}, V)$. Moreover, a TRS is called a* constructor TRS *if all its rewrite rules are constructor rules.*

Approximations are defined as follows, where it is *no longer* required for a rewrite rule $l \to r$ to satisfy $\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$:

Definition 2.2.5. *Let $\mathcal{R}$ and $\mathcal{S}$ be TRSs over the same signature. The TRS $\mathcal{S}$ is called an* approximation *of $\mathcal{R}$, if $\to_{\mathcal{R}}^* \subseteq \to_{\mathcal{S}}^*$ and $\mathsf{NF}_{\mathcal{R}} = \mathsf{NF}_{\mathcal{S}}$.*

Above, $\to_{\mathcal{R}}^*$ and $\to_{\mathcal{S}}^*$ denote the transitive-reflexive closures of $\to_{\mathcal{R}}$ and $\to_{\mathcal{S}}$, respectively. Observe, by definition of approximations, that each reduction to normal form in $\mathcal{R}$ can be 'mimicked' by $\mathcal{S}$. That is, for each reduction to normal form in $\mathcal{R}$ there exists a reduction to normal form in $\mathcal{S}$.

Example 2.2.6. Let Σ be a signature that consists of the nullary function symbols a and b and let $\mathcal{R} = (\Sigma, R)$ with $a \to b$ the sole element of R. The TRS $\mathcal{S} = (\Sigma, S)$, with $a \to x$ the sole element of S, is an approximation of $\mathcal{R}$. To see this, first remark that the only reduction possible in $\mathcal{R}$, i.e., $a \to b$, can be mimicked in $\mathcal{S}$ by defining $\sigma(x) = b$ and considering $\sigma(a) \to \sigma(x)$. Next, remark that all normal forms of $\mathcal{R}$, i.e., all variables and the function symbol b, are also normal forms of $\mathcal{S}$, becuase the left-hand side of the rewrite rule $a \to x$ applies neither to variables nor to b.

Additional details on approximations can be found in the papers by Jacquemard [Jac96] and by Durand and Middeldorp [DM97].

We next define descendants:

Definition 2.2.7. *Let $\mathcal{R} = (\Sigma, R)$ be an orthogonal TRS, $s, t \in \mathcal{T}er(\Sigma, V)$, and $p \in \mathcal{P}os(s)$. Let $s \to t$ be such that an $l \to r$-redex is contracted at position q in s. The set of* (static) descendants *of p across $s \to t$, denoted $p/(s \to t)$, is defined as:*

$$p/(s \to t) = \begin{cases} \{p\} & \text{if } p < q \text{ or } p \parallel q \\ \{q \cdot q_r \cdot q_p \mid r|_{q_r} = l|_{q_l}\} & \text{if } p = q \cdot q_l \cdot q_p \text{ with } l|_{q_l} \in V \\ \emptyset & \text{otherwise} \end{cases}$$

Let $s_0 \to^n s_n$ be a reduction of length n. The set of (static) descendants *of p across $s_0 \to^n s_n$, denoted $p/(s_0 \to^n s_n)$, is defined as:*

$$p/(s_0 \to^n s_n) = \begin{cases} \{p\} & \text{if } n = 0 \\ \bigcup_{q \in p/(s_0 \to^m s_m)} q/(s_m \to s_n) & \text{if } n = m + 1 \end{cases}$$

With the help of descendants, we can define inside-out reductions:

Definition 2.2.8 (Inside-Out). *Let $\mathcal{R} = (\Sigma, R)$ be an orthogonal TRS, and let*

$$s_0 \to_{p_1} s_1 \to_{p_2} \cdots \to_{p_i} s_i \to_{p_{i+1}} \cdots \to_{p_n} s_n$$

be a reduction such that the subscript p_i denotes the position of the redex contracted in $s_{i-1} \to s_i$. The reduction is called inside-out *if there do not exist $1 \leqslant j < k \leqslant n$ with a redex at a position $q > p_j$ in s_{j-1} such that $p_k \in q/(s_{j-1} \to^{k-j} s_{k-1})$.*

For all $n \in \mathbb{N}$, we denote by $s \to_{\mathrm{io}}^* t$, reductions $s \to^n t$ which are inside-out. With respect to inside-out reductions, Ariola [Ari96, Theorem 4.22] proves:

Lemma 2.2.9. *Let $\mathcal{R} = (\Sigma, R)$ be an orthogonal TRS and let $s, t \in \mathcal{T}er(\Sigma, V)$. If $s \to^* t$, then there exist $r \in \mathcal{T}er(\Sigma, V)$ such that $s \to^*_{\mathrm{io}} r \,{}^*\!\leftarrow t$ (see Figure 2.3).*

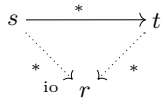

Figure 2.3. Lemma 2.2.9

2.3 Partial Orders

We next discuss the notions and notation needed with respect to partial orders. More details on partial orders can be found in the textbooks by Stoltenberg-Hansen, Lindström, and Griffor [SHLG94], by Davey and Priestley [DP02], and by Amadio and Curien [AC98].

By $\mathcal{P} = (P, \sqsubseteq)$ we denote a *partial order* $\sqsubseteq$ over a set P, i.e., a reflexive, transitive, and anti-symmetric relation over P. If $Q \subseteq P$, then Q is said to be *consistent*, if there exist $p \in P$ such that for all $q \in Q$ it holds that $q \sqsubseteq p$.

An element $p \in Q \subseteq P$ is called the *least element* of Q, if it is the unique element of Q such that for all $q \in Q$ it holds that $p \sqsubseteq q$. The element p is called the *greatest element* of Q, if it is the unique element of Q such that for all $q \in Q$ it holds that $q \sqsubseteq p$.

An element $p \in P$ is called the *least upper bound* of a set $Q \subseteq P$, if p is the least element of $\{q \in P \mid \forall r \in Q : r \sqsubseteq q\}$. Moreover, p is called the *greatest lower bound* of Q, if p is the greatest element of $\{q \in P \mid \forall r \in Q : q \sqsubseteq r\}$. The least upper bound of Q, respectively the greatest lower bound of Q, is denoted $\bigsqcup Q$, respectively $\bigsqcap Q$. In case $Q = \{p, q\}$ we also write $p \sqcup q$, respectively $p \sqcap q$.

The least upper bound of Q, respectively the greatest lower bound of Q, is called the *maximum* of Q, respectively the *minimum* of Q, if it is an element of Q. If it exists, we denote the maximum of Q, respectively the minimum of Q, by $\max Q$, respectively $\min Q$.

We call a non-empty set $D \subseteq P$ *directed*, if for all $p, q \in D$ there exist $r \in D$ 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 called a *Complete Partial Order (CPO)*, if P has a least element and if every directed subset of P has a least upper bound.

A partial order $\mathcal{P} = (P, \sqsubseteq)$ is called 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.

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:

$$\downarrow D = \{p \in P \mid \exists q \in D : p \sqsubseteq q\}\,.$$

Moreover, if $P^\infty = \{I \subseteq P \mid I \text{ is an ideal of } \mathcal{P}\}$ and if $\subseteq$ denotes subset inclusion, then $\mathcal{P}^\infty = (P^\infty, \subseteq)$ is a CPO. The partial order $\mathcal{P}^\infty$ is called the *ideal completion* of $\mathcal{P}$. The least upper bound of a directed set $Q \subseteq P^\infty$ is $\bigcup Q$. It must be remarked that the partial order $\mathcal{P}^\infty$ is actually a *domain*, see, e.g., Theorem 2.3 in the textbook by Stoltenberg-Hansen, Lindström, and Griffor [SHLG94]. We do not define domains here, as we do not make use of them.

2.4 Category Theory

In this section, we discuss the category theoretical notions and notation needed in Chapter 3. Most of the needed material is discussed in the textbook by Barr and Wells [BW99]. However, this textbook does not discuss coalgebras, which we also require. A good introduction to coalgebras is the tutorial on this topic by Jacobs and Rutten [JR97].

A *category* $\mathbf{C}$ is a collection of *objects* and *morphisms*. Each morphism has two associated objects called its *domain* and *codomain*. Given a morphism f with domain A and codomain B, we write $f : A \to B$. Each object A has an associated morphism $1_A : A \to A$ called its *identity*. Given morphisms $f : A \to B$ and $g : B \to C$, it is possible to *compose* f and g into a morphism $g \circ f : A \to C$. Composition must satisfy the following two requirements, given the morphisms f, g, and h with appropriate domains and codomains:

$$h \circ (g \circ f) = (h \circ g) \circ f$$
$$f \circ 1_A = f = 1_B \circ f$$

An example of a category is the *category of sets*, denoted **Set**, which has sets as objects and total maps between sets as morphisms. The composition of morphisms in the category of sets is the usual composition of total maps.

An *isomorphism* is a morphism $f : A \to B$, such that there exists a morphism $f^{-1} : B \to A$ for which it holds that:

$$f^{-1} \circ f = 1_A$$
$$f \circ f^{-1} = 1_B$$

It is a well-known fact that f^{-1} is unique.

A *final object*, denoted $\mathbf{1}$, is an object such that there exists for each object A a unique morphism with A as domain and the final object as codomain. Terminal objects are unique up to isomorphisms. In **Set** each singleton set is a terminal object. We employ the symbol $*$ to denote the unique member of the singleton set we choose to represent $\mathbf{1}$.

Given objects A and B, the *product* of A and B is defined as an object, denoted $A \times B$, together with a pair morphisms, denoted $\pi_1 : A \times B \to A$ and $\pi_2 : A \times B \to B$, such that there exists for each object C and for each pair of morphisms $f : C \to A$

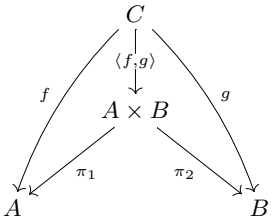

Figure 2.4. Product diagram

and $g : C \to B$ a unique morphism $\langle f, g\rangle$ that makes the diagram in Figure 2.4 commute.

Products, if they exist, are unique up to isomorphisms. The morphisms π_1 and π_2 are called *projections*. In **Set**, the product of two objects always exists and corresponds to the Cartesian product, as the notation already suggests.

Analogous to the product of a pair objects, we can define the product of n objects $A_1, \ldots, A_n$. We write $\prod_{i=1}^{n} A_i$ for the object of the product and π_i with $1 \leqslant i \leqslant n$ for the projections. As in the case of the product of two objects, products of n objects are unique up to isomorphisms. If $A_i = A$ for all $1 \leqslant i \leqslant n$, we write A^n, where $A^0 = \mathbf{1}$ and $A^1 = A$.

For notational convenience, we write $A \times B^n$ for a product $\prod_{i=1}^{n+1} C_i$ with $C_1 = A$ and $C_j = B$ for all $2 \leqslant j \leqslant n+1$. In addition, we write $A \times B^m \times C \times D^n$ for $\prod_{i=1}^{m+n+2} E_i$ where $E_1 = A$, $E_j = B$ for all $2 \leqslant j \leqslant m+1$, $E_{m+2} = C$, and $E_k = D$ for all $m+3 \leqslant k \leqslant m+n+2$.

Given objects A and B, the *sum* of A and B is defined as an object, denoted $A + B$, together with a pair morphisms, denoted $\mathrm{inl} : A \to A + B$ and $\mathrm{inr} : B \to A + B$, such that there exists for each object C and for each pair of morphisms $f : A \to C$ and $g : B \to C$ a unique morphism $[f, g]$ that makes the diagram in Figure 2.5 commute.

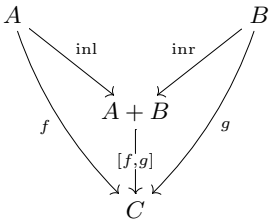

Figure 2.5. Sum diagram

Sums, if they exist, are unique up to isomorphisms. In **Set**, the sum of two objects always exists and corresponds to the disjoint union.

One particular example of a sum is the sum of the terminal object with itself, i.e., $\mathbf{1} + \mathbf{1}$. We denote $\mathbf{1} + \mathbf{1}$ by $\mathbf{2}$. In the case of **Set**, we have that $\mathbf{2}$ is a set with precisely two elements. We denote the two elements by 0 and 1.

Let I be some set, the sum of objects A_i for all $i \in I$ is defined analogous to the sum of a pair objects. We write $\coprod_{i \in I} A_i$ for the object of the sum and in_i with $i \in I$ for the morphisms. As in the case of the sum of two objects, sums of n objects are unique up to isomorphisms.

Let $\mathbf{C}$ and $\mathbf{D}$ be two categories. A *functor* F from $\mathbf{C}$ to $\mathbf{D}$ is a pair of maps, one from the objects of $\mathbf{C}$ to the objects of $\mathbf{D}$ and one from the morphisms of $\mathbf{C}$ to the morphisms of $\mathbf{D}$, both denoted F, such that:

– if $f : A \to B$ a morphism of $\mathbf{C}$, then $F(f) : F(A) \to F(B)$ a morphism of $\mathbf{D}$,
– $F(1_A) = 1_{F(A)}$ for all objects A of $\mathbf{C}$, and
– if $f \circ g$ a morphism of $\mathbf{C}$, then $F(f \circ g) = F(f) \circ F(g)$ a morphism of $\mathbf{D}$.

A functor F is called an *endofunctor*, if $\mathbf{D}$ is equal to $\mathbf{C}$. In this case we say that F is an endofunctor *over* $\mathbf{C}$.

Let F be an endofunctor over a category $\mathbf{C}$. The category of F-algebras is the category whose objects are pairs (A, α), called *F-algebras*, with A an object of $\mathbf{C}$ and $\alpha : F(A) \to A$, and whose morphisms, called *F-homomorphisms*, are morphisms of $\mathbf{C}$ such that f is a morphism from (A, α) to (B, β) if the following diagram commutes:

$$\begin{array}{ccc} F(A) & \xrightarrow{\alpha} & A \\ \downarrow{\scriptstyle F(f)} & & \downarrow{\scriptstyle f} \\ F(B) & \xrightarrow{\beta} & B \end{array}$$

An F-algebra is called *initial* if there exists a unique homomorphism to any other F-algebra. It is a well-known fact that the morphism that is part of an initial algebra is an isomorphism. In addition, initial algebras are unique up to isomorphisms.

Dual to F-algebras, the category of F-coalgebras is the category whose objects are pairs (A, α), called *F-coalgebras*, with A an object of $\mathbf{C}$ and $\alpha : A \to F(A)$, and whose morphisms, also called *homomorphisms*, are morphisms of $\mathbf{C}$ such that f is a morphism from (A, α) to (B, β) if the following diagram commutes:

$$\begin{array}{ccc} A & \xrightarrow{\alpha} & F(A) \\ \downarrow{\scriptstyle f} & & \downarrow{\scriptstyle F(f)} \\ B & \xrightarrow{\beta} & F(B) \end{array}$$

An F-coalgebra is called *final*, if there exists a unique homomorphism from any other F-coalgebra. It is a well-known fact that the morphism that is part of a final coalgebra is an isomorphism. In addition, final coalgebras are unique up to isomorphisms.

Let F be an endofunctor over **Set**. A *bisimulation* between F-coalgebras (A, α) and (B, β) is an F-coalgebra (R, ρ) together with a pair of homomorphisms $\pi_1 : (R, \rho) \to (A, \alpha)$ and $\pi_2 : (R, \rho) \to (B, \beta)$ such that $R \subseteq A \times B$.

It is a well-known fact that for each pair of homomorphisms $f : (A, \alpha) \to (B, \beta)$ and $g : (A, \alpha) \to (C, \gamma)$ it holds that $\{(f(a), g(a)) \mid a \in A\}$ is a bisimulation.

Moreover, for all coalgebras (A, α) and (B, β) there exists a greatest bisimulation, denoted $\sim$, which is the union of all other bisimulations between (A, α) and (B, β). If $(A, \alpha) = (B, \beta)$, then $\sim$ is a symmetric relation. Rutten [Rut00, Section 5] gives proofs of all these facts. A generalisation to categories other than **Set** can be found in Hughes' dissertation [Hug01].

3

Infinite Terms

Infinite: Bigger than the biggest thing ever and then some.
Much bigger than that in fact, really amazingly immense,
a totally stunning size, "wow, that's big", time.
Infinity is just so big that by comparison,
bigness itself looks really titchy.

— Douglas Adams
The Restaurant at the End of the Universe (1980)

In this chapter, we introduce *infinite terms*, sometimes called *trees* or *infinite trees*. Both infinite first-order terms and infinite λ-terms are introduced.

Infinite Terms. We can think of infinite terms as consisting of a root symbol and a *finite* number of subterms, where the number of subterms is equal to the arity of the root symbol and where the subterms are again infinite terms. As such, the process of recursively selecting subterms in an infinite term *may* continue forever. However, the process does not *necessarily* continue forever, as nullary function symbols and variables may occur.

The above implies that infinite terms differ from the terms defined in Chapter 2. Selecting subterms in those terms always ends after a finite number of steps, because the terms are finite. Of course, since selecting subterms of infinite terms does not necessarily continue forever, we may consider the set of terms defined in Chapter 2 to be a subset of the set of infinite terms.

Given that the set of (finite) terms defined in Chapter 2 can be considered to be a subset of the set of infinite terms, it might be said that the adjective 'infinite' in infinite terms somewhat inappropriate. Although this is the case, we stick with the adjective for two reasons:

– Using the adjective is common practice throughout the literature. Hence, it avoids confusion.
– In later chapters we need to be able to distinguish between the terms defined in Chapter 2 and the terms defined in the current chapter. The adjective provides a way to accomplish this goal.

We sometimes do omit the adjective if it is clear from the context that infinite terms are intended.

Coalgebras. Formally, we can define infinite terms as the *final coalgebra* of some functor. To understand why this is the case, one needs to be familiar with the usual interpretation of coalgebras in the category of sets and total functions. In this category, a morphism α of a coalgebra (A, α) is usually considered to be a map

that *deconstructs* the elements of A in their respective components. Hence, as we can think of infinite terms as *consisting* of their respective components, the root symbols and subterms, we are essentially dealing with a coalgebra. Requiring the coalgebra to be final makes sure that all infinite terms are represented.

Supposing that there are multiple representations of infinite terms, we obviously want to prove that the representations are isomorphic. A coalgebraic approach facilitate such a proof: If we show for all representations of infinite terms that they are final coalgebras of the same functor, then the result follows immediately, as final coalgebras are isomorphic (see Section 2.4).

Overview. We introduce infinite (first-order) terms in two steps. In the first step, as presented in Section 3.1, we define infinite terms as a *final coalgebra* of a certain functor. We do not give a concrete set theoretic representation of the final coalgebra. However, we do provide, for infinite terms, categorical definitions of *positions*, *subterms*, and *replacements of subterms*. That is, we define the relevant maps in a categorical fashion.

In the second step, which encompasses Sections 3.2, 3.3, and 3.4, we discuss three representations of infinite terms that occur throughout the literature. The three representations can be summarised as *ideal completion*, *partial functions*, and *metric completion*. In all three cases we show that a final coalgebra for the functor introduced in Section 3.1 is defined. We also show that the usual definitions of positions, subterms, and replacements of subterms that exist for the three representations coincide with the categorical definitions.

In Section 3.5, we introduce infinite λ-terms, following the same two steps as in the case of infinite first-order terms. Infinite λ-terms are required in Chapter 4, where the Böhm-like trees of the λ-calculus are surveyed.

It must be noted that is not required for the remaining chapters of this dissertation to have knowledge of all material presented in this chapter. Only in Chapter 4 is knowledge of most of the material required. In Chapters 5 through 9, with the exception of Chapter 7, only knowledge of Sections 3.2.1 and 3.2.2 is required. In Chapter 7, knowledge of Sections 3.2.1, 3.2.2, and 3.4.2 is required.

Throughout this chapter we assume that Σ is an arbitrary signature and that V is a countably infinite set of variables. Moreover, to explain the definitions we make use of the signature of Combinatory Logic (CL). That is, the signature defined as $\Sigma_{\mathrm{CL}} = \{S, K, I, \cdot\}$ with S, K, and I nullary function symbols and $\cdot$ a binary function symbol. As usual, given terms s and t, we shall write $s\,t$ instead of $\cdot(s, t)$. Moreover, assuming left-associativity, parentheses are omitted accordingly. For more details on CL see, e.g., Barendregt's book [Bar84] or the book by Terese [Ter03].

3.1 Coalgebraic Definitions

In Section 3.1.1 we define the functor whose final coalgebra represents infinite terms. In the same section we also define a functor that has as its final coalgebra all sets of strings over $\mathbb{N}$, i.e., $\wp(\mathbb{N}^*)$. Strings over $\mathbb{N}$ are needed to define positions of infinite terms.

In Sections 3.1.2, 3.1.3, and 3.1.4, we give categorical definitions of *positions*, *subterms*, and *replacements of subterms* that apply to infinite terms. In the definitions, the functors defined in Section 3.1.1 are employed.

Throughout the current section we assume we are working with **Set**, the category of sets and total maps. Doing so avoids the need to generalise the concepts of signatures and variables. Generalising the concepts is very well possible, as shown, e.g., by Robinson [Rob94] and Gahni, Lüth, De Marchi, and Power [GLMP03].

Working with **Set** also allows for some notational conveniences. For example, given a set X, we can write $x \in X$ instead of $x : \mathbf{1} \to X$. Moreover, given sets X and Y such that $X \cap Y = \emptyset$, we can differentiate between the elements of $X + Y$, as coming from X and Y, by writing $x \in X$ and $y \in Y$.

3.1.1 Infinite Terms and Sets of Positions

Infinite Terms. As is well-known, see, e.g., Turi's dissertation [Tur96], it is possible to define $\mathcal{T}er(\Sigma, V)$, i.e., the set of (finite) terms over Σ and V, as the initial algebra of the *term functor*:

$$F_\Sigma(X) = V + \coprod_{n \in \mathbb{N}} \Sigma_n \times X^n \,,$$

which is obviously an endofunctor over **Set**.

Example 3.1.1. Given the signature Σ_{CL}, the term functor becomes:

$$F_{\Sigma_{\mathrm{CL}}}(X) = V + \{S, K, I\} \times \mathbf{1} + \{\cdot\} \times X^2 \,,$$

where $\{S, K, I\} \times \mathbf{1}$ is actually $\{S, K, I\} \times X^0$. We denote the elements of $\{S, K, I\} \times \mathbf{1}$ by S, K, and I instead of by $(S, *)$, $(K, *)$, and $(I, *)$, for the second member of each pair is always $*$.

Given an endofunctor, it is well-known, see, e.g., the tutorial by Jacobs and Rutten [JR97], that 'infinite structures' are obtained through dualisation. That is, by considering final coalgebras instead of initial algebras. For this reason, we define the set of infinite terms as the final coalgebra of the term functor.

We denote the final F_Σ-coalgebra by:

$$(\mathcal{T}er^\infty(\Sigma, V), \varphi_\infty) \,,$$

where φ_∞ is a map from $\mathcal{T}er^\infty(\Sigma, V)$ to $F_\Sigma(\mathcal{T}er^\infty(\Sigma, V))$. In Sections 3.2, 3.3, 3.4 we discuss three representations of the final coalgebra.

To avoid notational clutter, we usually denote $(\mathcal{T}er^\infty(\Sigma, V), \varphi_\infty)$ plainly by $\mathcal{T}er^\infty(\Sigma, V)$. Moreover, by S, T, ... we denote arbitrary elements of $\mathcal{T}er^\infty(\Sigma, V)$. This allows us to keep them apart from the elements of $\mathcal{T}er(\Sigma, V)$, which we denote by s, t, ... (see Section 2.2.1).

Recall that a morphism α of a coalgebra (A, α) is usually considered to be a map that *deconstructs* the elements of A in their respective components. As such, there is a very clear interpretation of the map φ_∞. If we apply φ_∞ to an infinite

term which is not a variable, then we obtain the root symbol $f \in \Sigma_n$ and the n subterms, i.e., we obtain an element of $\Sigma_n \times \mathcal{T}er^\infty(\Sigma, V)^n$. Otherwise, if we apply φ_∞ to a variable, then we obtain that variable, i.e., an element of V.

As we shall see, the concrete definitions of φ_∞ as given in Sections 3.2, 3.3, and 3.4 agree with above description.

Remark that nothing in the above interpretation of φ_∞ forbids the occurrence of a certain function symbol at the root of an infinite term obtained by repeatedly applying φ_∞ and selecting one of the acquired subterms. Hence, the elements of $\mathcal{T}er^\infty(\Sigma, V)$ can actually represent infinite objects: There is *no* reason why the process of repeatedly applying φ_∞ and selecting subterms should terminate.

Example 3.1.2. Assume I^ω denotes the infinite term that is the unique solution of the following equation:

$$x = Ix\,.$$

We have:

$$\varphi_\infty(KI^\omega) = (\cdot, K, I^\omega)\,.$$

Hence, $\cdot$ is the root symbol and K and I^ω are its two subterms. We also have:

$$\begin{aligned} \pi_1 \circ \varphi_\infty(KI^\omega) &= \cdot \\ \pi_2 \circ \varphi_\infty(KI^\omega) &= K \\ \varphi_\infty \circ \pi_3 \circ \varphi_\infty(KI^\omega) &= (\cdot, I, I^\omega) \end{aligned}$$

The term I^ω can be defined in a coalgebraic fashion. To see this, consider the following commutative diagram:

$$\begin{array}{ccc} \mathbf{1} & \xrightarrow{\quad\varphi_{\mathbf{1}}\quad} & F_{\Sigma_{\mathrm{CL}}}(\mathbf{1}) \\ \downarrow{\scriptstyle m} & & \downarrow{\scriptstyle F_{\Sigma_{\mathrm{CL}}}(m)} \\ \mathcal{T}er^\infty(\Sigma_{\mathrm{CL}}, V) & \xrightarrow{\quad\varphi_\infty\quad} & F_{\Sigma_{\mathrm{CL}}}(\mathcal{T}er^\infty(\Sigma_{\mathrm{CL}}, V)) \end{array}$$

where $\varphi_{\mathbf{1}}$ is defined as:

$$\varphi_{\mathbf{1}}(*) = (\cdot, I, *)\,.$$

When thinking of $*$ as representing the variable x, it is obvious we can think of $\varphi_{\mathbf{1}}$ as representing the recursive equation.

Because $\mathcal{T}er^\infty(\Sigma_{\mathrm{CL}}, V)$ is a final coalgebra, there exists a unique homomorphism m that makes the above diagram commute. Since m has $\mathbf{1}$ as its domain, it singles out a unique element of $\mathcal{T}er^\infty(\Sigma_{\mathrm{CL}}, V)$. Whence, m can be seen the unique solution to the recursive equation and it thus defines I^ω.

We do not give a formal treatment of recursive equations, like the one in the above example, as we do not use them in the remaining chapters of this dissertation. However, it should be noted that commutative diagrams similar to the one above are employed below to define positions, subterms, and replacements.

To finalise our discussion of infinite terms, as far as this section goes, we give a concrete definition of a bisimulation for F_Σ-coalgebras. The definition is easily seen to instantiate the general definition given in Chapter 2.

Definition 3.1.3. *Let (A, α) and (B, β) be F_Σ-coalgebras.* A bisimulation *between (A, α) and (B, β) is a relation $R \subseteq A \times B$ such that $a\ R\ b$ implies:*

1. *$\alpha(a), \beta(b) \in V$ and $\alpha(a) = \beta(b)$, or*
2. *$\alpha(a) \in \Sigma_n \times A^n$ and $\beta(b) \in \Sigma_n \times B^n$ for some $n \in \mathbb{N}$ and:*
   - *$\pi_1 \circ \alpha(a) = \pi_1 \circ \beta(b)$, and*
   - *$\pi_{i+1} \circ \alpha(a)\ R\ \pi_{i+1} \circ \beta(b)$ for all $1 \leqslant i \leqslant n$.*

Sets of Positions. As in the case of (finite) terms, we denote the positions of infinite terms by strings over $\mathbb{N}$. We do not need strings of *infinite* length: Between the root of an infinite term and each of its function symbols, there can only be a finite number of other function symbols. Of course, since an infinite term can be infinitely large, the *set* of all positions of an infinite term can be infinite.

Subsets of $\mathbb{N}^*$, as described above, are of course elements of $\wp(\mathbb{N}^*)$. Since we want to define, in Section 3.1.2, a homomorphism from infinite terms to $\wp(\mathbb{N}^*)$ in a fashion similar to I^ω in Example 3.1.2, we need to define $\wp(\mathbb{N}^*)$ as the final coalgebra of some functor. Rutten [Rut98, Rut00] provides a solution to this problem.

Given a set A, Rutten defines the functor:

$$D_A(X) = \mathbf{2} \times X^A \,.$$

All D_A-coalgebras can be viewed as *deterministic automata* with A as input alphabet. That is, D_A-coalgebras are pairs $(S, \langle \mathrm{o}, \mathrm{t} \rangle)$, with S a set of *states*, $\mathrm{o} : S \to \mathbf{2}$ an *output function*, and $\mathrm{t} : S \to S^A$ a *transition function*. With regard to o, we assume that a state $x \in S$ is *accepting* if $\mathrm{o}(x) = 1$ and that it is *non-accepting* otherwise.

A final D_A-coalgebra is the deterministic automaton $(\wp(A^*), \langle \mathrm{o}_A, \mathrm{t}_A \rangle)$, with o_A defined as:

$$\mathrm{o}_A(X) = \begin{cases} 0 & \text{if } \epsilon \notin X \\ 1 & \text{if } \epsilon \in X \end{cases}$$

and, with t_A defined, for each $a \in A$, as:

$$\mathrm{t}_A(X)(a) = X|_a \,,$$

where $X|_a$ denotes the set of suffixes of X with respect to a.

In case of $A = \mathbb{N}$, the final coalgebra specialises to:

$$(\wp(\mathbb{N}^*), \langle \mathrm{o}_\mathbb{N}, \mathrm{t}_\mathbb{N} \rangle) \,.$$

Hence, we have a final coalgebra representing $\wp(\mathbb{N}^*)$, as we required.

To see how we can employ the final coalgebra to represent some subset of $\mathbb{N}^*$, consider the following example:

Example 3.1.4. Suppose we want to represent the set $\{\epsilon, 1, 2, 21\} \subseteq \mathbb{N}^*$. To do so, assume $\mathbf{3} = \{a, b, c\}$ and consider the following commutative diagram:

$$\begin{array}{ccc} \mathbf{1} + \mathbf{3} & \xrightarrow{\langle \mathrm{o}_\mathbf{3}, \mathrm{t}_\mathbf{3} \rangle} & D_\mathbb{N}(\mathbf{1} + \mathbf{3}) \\ \downarrow{\scriptstyle m} & & \downarrow{\scriptstyle D_\mathbb{N}(m)} \\ \wp(\mathbb{N}^*) & \xrightarrow{\langle \mathrm{o}_\mathbb{N}, \mathrm{t}_\mathbb{N} \rangle} & D_\mathbb{N}(\wp(\mathbb{N}^*)) \end{array}$$

with $\mathrm{o}_{\mathbf{3}}$ defined as:

$$\mathrm{o}_{\mathbf{3}}(x) = \begin{cases} 0 & \text{if } x \in \mathbf{1} \\ 1 & \text{if } x \in \mathbf{3} \end{cases}$$

and with $\mathrm{t}_{\mathbf{3}}$ defined as:

$$\mathrm{t}_{\mathbf{3}}(x)(i) = \begin{cases} * & \text{if } x = * \\ b & \text{if } x = a \text{ and } i = 1 \\ c & \text{if } x = a \text{ and } i = 2 \\ * & \text{if } x = a \text{ and } i \neq 1 \text{ and } i \neq 2 \\ * & \text{if } x = b \\ b & \text{if } x = c \text{ and } i = 1 \\ * & \text{if } x = c \text{ and } i \neq 1 \end{cases}$$

The pair $(\mathbf{1}+\mathbf{3}, \langle \mathrm{o}_{\mathbf{3}}, \mathrm{t}_{\mathbf{3}} \rangle)$ represents the deterministic automaton depicted in Figure 3.1. The states a, b, and c are *accepting states* and the state $*$ is a *trap state* (see, e.g., Linz' book [Lin96]). For each state, the outgoing arrows are labelled with pairwise disjoint subsets of $\mathbb{N}$. A transition along one of the arrows is possible, whenever one of the elements in its label is encountered.

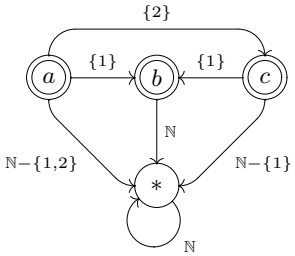

Figure 3.1. Automaton accepting $\{\epsilon, 1, 2, 21\}$ from state a

Since $\wp(\mathbb{N}^*)$ is a final coalgebra, there exists a unique homomorphism m from $\mathbf{1} + \mathbf{3}$ to $\wp(\mathbb{N}^*)$. It is readily shown that m is the following map:

$$m(x) = \begin{cases} \emptyset & \text{if } x = * \\ \{\epsilon, 1, 2, 21\} & \text{if } x = a \\ \{\epsilon\} & \text{if } x = b \\ \{\epsilon, 1\} & \text{if } x = c \end{cases}$$

Hence, given the map $a : \mathbf{1} \to \mathbf{3}$ defined by $a(*) = a$, we can represent $\{\epsilon, 1, 2, 21\}$ by $m \circ a(*)$.

Remark 3.1.5. As is easily inferred from the definition in the next section, not every subset of $\mathbb{N}^*$ corresponds to the set of positions of some term. The sets obtained

are always downward closed with respect to the prefix order on strings. The use of $\wp(\mathbb{N}^*)$ stems from the wish to be explicit about the fact that each set of positions is a subset of $\mathbb{N}^*$.

3.1.2 Positions

In this section, we define the morphism $\mathcal{P}os : \mathcal{T}er^\infty(\Sigma, V) \to \wp(\mathbb{N}^*)$ that maps each infinite term to the sets of its positions. Given an infinite term S, this means that $\mathcal{P}os$ maps S to the set that contains both ϵ and $i \cdot p$ for all subterms $\pi_{i+1} \circ \varphi_\infty(S)$ of S and positions $p \in \mathcal{P}os(\pi_{i+1} \circ \varphi_\infty(S))$. This is similar to the map $\mathcal{P}os$ defined for (finite) terms in Section 2.2.1.

To define $\mathcal{P}os$, we mimic Example 3.1.4, where $\mathbf{3}$ is replaced by $\mathcal{T}er^\infty(\Sigma, V)$. As such, we first define a deterministic automaton on $\mathbf{1} + \mathcal{T}er^\infty(\Sigma, V)$, which implies the existence of a homomorphism $\mathcal{P}os' : \mathbf{1} + \mathcal{T}er^\infty(\Sigma, V) \to \wp(\mathbb{N}^*)$. Thereafter, $\mathcal{P}os$ is defined based on $\mathcal{P}os'$.

In analogy to Example 3.1.4, we employ the unique element $* \in \mathbf{1}$ as a trap state. The trap state is used to deal with two phenomena:

- If a variable occurs at the root, then there are no subterms. This means ϵ is the only position. Hence, only ϵ should be accepted and the trap state should be reached in any other case.
- If a function symbol of arity n occurs at the root of S, then there are no subterms other than $\pi_{i+1} \circ \varphi_\infty(S)$ for all $1 \leqslant i \leqslant n$. Hence, besides accepting ϵ the only transitions to other accepting states should be to states accepting the positions of the subterms $\pi_{i+1} \circ \varphi_\infty(S)$ for all $1 \leqslant i \leqslant n$. Once again, the trap state should be reached in any other case.

Hence, the trap state is reached in case a certain string over $\mathbb{N}$ does not correspond to a position of a subterm. Moreover, any other state, i.e., any infinite term, is an accepting state.

To define $\mathcal{P}os'$, consider the following commutative diagram:

$$\begin{array}{ccc} \mathbf{1} + \mathcal{T}er^\infty(\Sigma, V) & \xrightarrow{\langle \mathrm{o_p}, \mathrm{t_p} \rangle} & D_{\mathbb{N}}(\mathbf{1} + \mathcal{T}er^\infty(\Sigma, V)) \\ \downarrow{\scriptstyle \mathcal{P}os'} & & \downarrow{\scriptstyle D_{\mathbb{N}}(\mathcal{P}os')} \\ \wp(\mathbb{N}^*) & \xrightarrow{\langle \mathrm{o_N}, \mathrm{t_N} \rangle} & D_{\mathbb{N}}(\wp(\mathbb{N}^*)) \end{array}$$

where $\mathrm{o_p}$ is defined as:

$$\mathrm{o_p}(x) = \begin{cases} 0 & \text{if } x \in \mathbf{1} \\ 1 & \text{if } x \in \mathcal{T}er^\infty(\Sigma, V) \end{cases}$$

and where $\mathrm{t_p} = [\mathrm{t_p^{\mathbf{1}}}, \mathrm{t_p^{\mathcal{T}}}]$ with:

$$\mathrm{t_p^{\mathbf{1}}}(x)(i) = *$$

$$\mathrm{t}_{\mathrm{p}}^{\mathcal{T}}(x)(i) = \begin{cases} * & \text{if } \varphi_\infty(x) \in V \\ * & \text{if } \pi_1 \circ \varphi_\infty(x) \in \Sigma_n \text{ with } i > n \\ \pi_{i+1} \circ \varphi_\infty(x) & \text{if } \pi_1 \circ \varphi_\infty(x) \in \Sigma_n \text{ with } 1 \leqslant i \leqslant n \end{cases}$$

Since $\wp(\mathbb{N}^*)$ is a final $D_{\mathbb{N}}$-coalgebra, there exists a unique homomorphism $\mathcal{P}os'$ from $\mathbf{1}+\mathcal{T}er^\infty(\Sigma, V)$ to $\wp(\mathbb{N}^*)$. Hence, we can define the following map from $\mathcal{T}er^\infty(\Sigma, V)$ to $\wp(\mathbb{N}^*)$, which we denote by $\mathcal{P}os$:

$$\mathcal{P}os(S) = \mathcal{P}os' \circ \mathrm{inr}(S)\,,$$

where $S \in \mathcal{T}er^\infty(\Sigma, V)$.

By definition of $\langle \mathrm{o_p}, \mathrm{t_p} \rangle$, it is fairly easy to see that $\mathcal{P}os$ satisfies the requirements given at the beginning of this section. However, to obtain some insight in why $\mathcal{P}os$ actually defines the set of positions of an infinite term, consider the following example:

Example 3.1.6. In case of $I^\omega \in \mathcal{T}er^\infty(\Sigma_{\mathrm{CL}}, V)$, as defined in Example 3.1.2, it is obvious that we should have:

$$\mathcal{P}os(I^\omega) = \{2^n, 2^n \cdot 1 \mid n \in \mathbb{N}\}\,,$$

where $2^0 = \epsilon$, $2^1 = 2$, and $2^2 = 22$, etc. That the equality actually holds, is straightforwardly checked with the help of Figure 3.2. The figure depicts a fragment of the automaton accepting $\mathcal{P}os'(x)$ for each $x \in \mathbf{1}+\mathcal{T}er^\infty(\Sigma_{\mathrm{CL}}, V)$, as follows easily by inspection of the definition of $\langle \mathrm{o_p}, \mathrm{t_p} \rangle$.

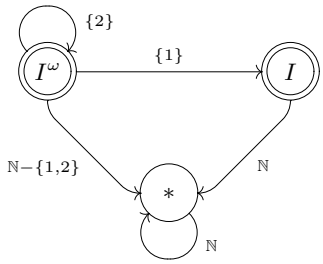

Figure 3.2. Fragment of the automaton accepting $\mathcal{P}os'(x)$

3.1.3 Subterms

We next define for each $S \in \mathcal{T}er^\infty(\Sigma, V)$ and $p \in \mathcal{P}os(S)$ the subterm at position p in S. That is, we define a morphism from $\mathcal{TP}$ to $\mathcal{T}er^\infty(\Sigma, V)$, where:

$$\mathcal{TP} = \{(S, p) \mid S \in \mathcal{T}er^\infty(\Sigma, V) \text{ and } p \in \mathcal{P}os(S)\}\,,$$

a subset of $\mathcal{T}er^\infty(\Sigma, V) \times \wp(\mathbb{N}^*)$. We denote the morphism by $\_|\_$.

Analogous to the definition of $\_|\_$ for (finite) terms (see Chapter 2), we want the subterm at position p in S to be equal to S whenever $p = \epsilon$ and we want it to be equal to $(\pi_{i+1} \circ \varphi_\infty(S))|_q$ whenever $p = i \cdot q$. In the second case, $\pi_{i+1} \circ \varphi_\infty(S)$ exists, since $q \cdot i \in \mathcal{P}os(S)$ implies $1 \leqslant i \leqslant n$ with $\pi_1 \circ \varphi_\infty(S) \in \Sigma_n$ by definition of $\langle \mathrm{o_p}, \mathrm{t_p} \rangle$.

Remark that the above explanation is in terms of positions, which are defined by means of an initial algebra. As such, we also employ initiality in the definition of $\_|\_$. That is, we proceed in a fashion that is dual to that of previous section.

The initial algebra we employ is not the initial algebra of positions. It is an initial algebra that defines $\mathcal{TP}$. To see that $\mathcal{TP}$ can actually be defined by initiality, first consider for all $n \in \mathbb{N}$ and $1 \leqslant i \leqslant n$ the endofunctor TP_i^n defined as:

$$TP_i^n(X) = \Sigma_n \times \mathcal{T}er^\infty(\Sigma, V)^{i-1} \times X \times \mathcal{T}er^\infty(\Sigma, V)^{n-i}$$

and define:

$$TP(X) = \mathcal{T}er^\infty(\Sigma, V) + \coprod_{n \in \mathbb{N} - \{0\}} \left(\coprod_{1 \leqslant i \leqslant n} TP_i^n(X) \right).$$

Now, consider the pair:

$$(\mathcal{TP}, \psi_{\mathrm{tp}}),$$

where ψ_{tp} is the map from $TP(\mathcal{TP})$ to $\mathcal{TP}$ defined as:

$$\psi_{\mathrm{tp}}(x) = \begin{cases} (x, \epsilon) & \text{if } x \in \mathcal{T}er^\infty(\Sigma, V) \\ (S, i \cdot p) & \text{if } x \in TP_i^n(\mathcal{TP}) \text{ with } \pi_1 \circ \varphi(S) = \pi_1(x), \\ & \quad (\pi_{i+1} \circ \varphi_\infty(S), p) = \pi_{i+1}(x), \text{ and} \\ & \quad \pi_{j+1} \circ \varphi_\infty(S) = \pi_{j+1}(x) \text{ for all } j \neq i \end{cases}$$

That the infinite term S in the pair $(S, i \cdot p)$ actually exists follows by induction on the length of $i \cdot p$, employing in the induction step a construction similar to the one in Example 3.1.2.

As required, the pair $(\mathcal{TP}, \psi_{\mathrm{tp}})$ is an initial algebra of the endofunctor TP. This follows easily by induction on the length of the positions that occur in the pairs of $\mathcal{TP}$.

To define $\_|\_$, we first construct a homomorphism, which we denote $\_|'\_$. Employing the initial algebra just defined, we construct an *algebra* homomorphism. This is different from what we did in the case of $\mathcal{P}os$, which was defined by means of a *coalgebra* homomorphism.

Consider the following commutative diagram:

$$\begin{array}{ccc} TP(\mathcal{TP}) & \xrightarrow{\quad\psi_{\mathrm{tp}}\quad} & \mathcal{TP} \\ \downarrow{\scriptstyle TP(\_|'\_)} & & \downarrow{\scriptstyle \_|'\_} \\ TP(\mathcal{T}er^\infty(\Sigma, V) \times \mathcal{TP}) & \xrightarrow{\quad\psi_{\mathrm{s}}\quad} & \mathcal{T}er^\infty(\Sigma, V) \times \mathcal{TP} \end{array}$$

where ψ_s is defined as:

$$\psi_s(x) = \begin{cases} (x, (x, \epsilon)) & \text{if } x \in \mathcal{T}er^\infty(\Sigma, V) \\ (S, (T, i \cdot p)) & \text{if } x \in TP_i^n(\mathcal{T}er^\infty(\Sigma, V) \times \mathcal{TP}) \text{ with} \\ & \quad \pi_1 \circ \varphi_\infty(T) = \pi_1(x), \\ & \quad (S, (\pi_{i+1} \circ \varphi_\infty(T), p)) = \pi_{i+1}(x), \text{ and} \\ & \quad \pi_{j+1} \circ \varphi_\infty(T) = \pi_{j+1}(x) \text{ for all } j \neq i \end{cases}$$

That the infinite term T in the pair $(T, i \cdot p)$ actually exists follows by induction on the length of $i \cdot p$, employing in the induction step a construction similar to the one in Example 3.1.2.

By initiality of $\mathcal{TP}$, there exists a unique homomorphism $\_|'\_$ from $\mathcal{TP}$ to $\mathcal{T}er^\infty(\Sigma, V) \times \mathcal{TP}$. Hence, we can define a map from $\mathcal{TP}$ to $\mathcal{T}er^\infty(\Sigma, V)$, which we denote by $\_|\_$:

$$S|_p = \_|\_(S, p) = \pi_1 \circ \_|'\_(S, p)\,,$$

where $(S, p) \in \mathcal{TP}$. By definition of $\_|'\_$, it is fairly easy to see that $\_|\_$ satisfies the requirements given at the beginning of this section.

Remark 3.1.7. If we would categorically define subterms of (finite) terms, then a definition can be given that is almost identical to the one above. We would only have to replace each occurrence of $\mathcal{T}er^\infty(\Sigma, V)$ by $\mathcal{T}er(\Sigma, V)$.

We end this section with an example:

Example 3.1.8. Given the term $KI^\omega \in \mathcal{T}er^\infty(\Sigma, V)$ from Example 3.1.2, we have the following set of positions:

$$\mathcal{P}os(KI^\omega) = \{2^n, 2^n \cdot 1 \mid n \in \mathbb{N}\}\,,$$

as is easily deduced from Example 3.1.6. Hence, we also have:

$$\begin{gathered} KI^\omega|_\epsilon = KI^\omega \\ KI^\omega|_1 = K|_\epsilon = K \\ KI^\omega|_2 = I^\omega|_\epsilon = I^\omega \\ KI^\omega|_{21} = I^\omega|_1 = I|_\epsilon = I \\ KI^\omega|_{22} = I^\omega|_2 = I^\omega|_\epsilon = I^\omega \end{gathered}$$

3.1.4 Replacements of Subterms

Given infinite terms S and T and a position $p \in \mathcal{P}os(S)$, we next define the replacement of the subterm in S at position p by T. That is, we define a morphism from $\mathcal{TPT}$ to $\mathcal{T}er^\infty(\Sigma, V)$, where:

$$\mathcal{TPT} = \{(S, p, T) \mid S, T \in \mathcal{T}er^\infty(\Sigma, V) \text{ and } p \in \mathcal{P}os(S)\}\,.$$

We denote the morphism by $\_[\_]\_$.

As in the case of replacements in (finite) terms (see Chapter 2), we want $S[T]_\epsilon = T$. Moreover, we want $S[T]_{i\cdot p} = (f, S_1, \ldots, S_n)$ if $\pi_1 \circ \varphi_\infty(S) = f \in \Sigma_n$, $S_i = (\pi_{i+1} \circ \varphi_\infty(S))[T]_p$, and $S_j = \pi_{j+1} \circ \varphi_\infty(S)$ for all $j \neq i$.

In analogy to the previous sections, we proceed by first defining a homomorphism $\_[\_]'\_$. Consider the following commutative diagram:

$$\begin{array}{ccc} \mathcal{T}er^\infty(\Sigma, V) + \mathcal{TPT} & \xrightarrow{\varphi_{\mathrm{r}}} & F_\Sigma(\mathcal{T}er^\infty(\Sigma, V) + \mathcal{TPT}) \\ \downarrow{\scriptstyle \_[\_]'\_} & & \downarrow{\scriptstyle F_\Sigma(\_[\_]'\_)} \\ \mathcal{T}er^\infty(\Sigma, V) & \xrightarrow{\varphi_\infty} & F_\Sigma(\mathcal{T}er^\infty(\Sigma, V)) \end{array}$$

and define φ_{r} as:

$$\varphi_{\mathrm{r}}(x) = \begin{cases} F_\Sigma(\mathrm{inl}) \circ \varphi_\infty(x) & \text{if } x \in \mathcal{T}er^\infty(\Sigma, V) \\ F_\Sigma(\mathrm{inl}) \circ \varphi_\infty(T) & \text{if } x = (S, \epsilon, T) \in \mathcal{TPT} \\ (f, y_1, \ldots, y_n) & \text{if } x = (S, i \cdot p, T) \in \mathcal{TPT} \text{ with} \\ & \quad \pi_1 \circ \varphi_\infty(S) = f \in \Sigma_n, \\ & \quad y_i = (\pi_{i+1} \circ \varphi_\infty(S), p, T), \text{ and} \\ & \quad y_j = \pi_{j+1} \circ \varphi_\infty(S) \text{ for all } j \neq i \end{cases}$$

By finality of $\mathcal{T}er^\infty(\Sigma, V)$, there exists a unique homomorphism $\_[\_]'\_$ from $\mathcal{TPT} + \mathcal{T}er^\infty(\Sigma, V)$ to $\mathcal{T}er^\infty(\Sigma, V)$. Hence, we can define the following map from $\mathcal{TPT}$ to $\mathcal{T}er^\infty(\Sigma, V)$, which we denote by $\_[\_]\_$:

$$S[T]_p = \_[\_]\_(S, p, T) = \_[\_]'\_ \circ \mathrm{inr}(S, p, T)\,,$$

where $(S, p, T) \in \mathcal{TPT}$. By definition of $\_[\_]'\_$, it is fairly easy to see that $\_[\_]\_$ satisfies the requirements given at the beginning of this section.

To gain some more insight in the reason why $\_[\_]\_$ is defined as expected, consider the following example:

Example 3.1.9. Given the nullary function symbol K, the positions $\epsilon, 1, 2, 22 \in \mathcal{P}os(S)$, and the infinite term I^ω, as defined in Example 3.1.2, we obtain:

$$\begin{gathered} I^\omega[K]_\epsilon = I^\omega[K]'_\epsilon = K \\ I^\omega[K]_1 = I^\omega[K]'_1 = (I[K]'_\epsilon)I^\omega = KI^\omega \\ I^\omega[K]_2 = I^\omega[K]'_2 = I(I^\omega[K]'_\epsilon) = IK \\ I^\omega[K]_{22} = I^\omega[K]'_{22} = I(I^\omega[K]'_2) = I(IK) \end{gathered}$$

3.2 Ideal Completion†

Given a term s, a *prefix* of s is obtained by omitting zero or more subterms from s. That is, a number of subterms of s is left unspecified. Obviously, any (finite) term

†This section is partially based on earlier work by the author [Ket04].

can be represented by the set of all its prefixes. Moreover, ordering the prefixes based on the subterms that are left unspecified, three properties are readily proved with respect to the set containing all prefixes of a (finite) term:

1. the set is finite,
2. the set is downward closed, and
3. for each pair of elements in the set, the least upper bound is also in the set.

Hence, the set containing all of prefixes of a (finite) term is a *finite ideal*.

The above observations provide the idea for the representation of infinite terms that we discuss in this section. That is, to represent the infinite terms as *finite* and *infinite* ideals over the set of terms. In other words, to use *ideal completion*.

The prefix relation and infinite terms are defined respectively in Sections 3.2.1 and 3.2.2. In Sections 3.2.3 and 3.2.4, we prove respectively that ideal completion defines a final F_Σ-coalgebra and that the morphisms defined in Section 3.1 correspond to the ones usually defined in the case of ideal completion.

Bibliographic Notes. A number of publications on denotational semantics of recursive program schemes employ ideal completion to define infinite terms, e.g., the paper by Berry and Lévy [BL79]. In the context of TRSs, infinite terms defined by means of ideal completion are employed, e.g., by Ariola [Ari96]. In each case, there are strong connections with domains as defined by Scott [Sco72, Sco76].

3.2.1 Partial Terms

To represent unspecified subterms we extend the signature Σ with a fresh nullary function symbol $\bot$. That is, we extend the signature with a function symbol that occurs neither in Σ nor in V. We call the subterms that are equal to $\bot$ the *unspecified subterms*. Moreover, we call the set of terms $\mathcal{T}er(\Sigma_\bot, V)$ over the signature $\Sigma_\bot = \Sigma \cup \{\bot\}$ the set of *partial terms*. We drop the adjective partial if it is obvious from the context that partial terms are considered.

With the help of $\bot$ we can define two prefix orders on terms, where a term s is understood to be a prefix of a term t when there exist unspecified subterms in s that are specified in t, but not the other way around (see Figure 3.3 for a graphic representation).

Definition 3.2.1. *Let Σ be a signature and V a set of variables.*

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

$$s \prec t \iff (s \preccurlyeq t \text{ and } t \not\preccurlyeq s)\,.$$

Figure 3.3. Prefix order on $\mathcal{T}er(\Sigma_\bot, V)$

If we have $s \preccurlyeq t$, respectively $s \prec t$, then we call s a *prefix* of t, respectively a *strict prefix* of t. Moreover, by $\succcurlyeq$ and $\succ$ we denote respectively the converse of the prefix order and the converse of the strict prefix order.

Example 3.2.2. In the case of $\mathcal{T}er(\Sigma_{\mathrm{CL},\bot}, V)$ we have:

$$\bot \prec I \qquad\qquad I\bot \prec II$$
$$\bot\bot \preccurlyeq \bot\bot \qquad\qquad IK \not\preccurlyeq II$$

With respect to the prefix order and the strict prefix order, the subsequent lemma holds:

Lemma 3.2.3. *The pairs $\mathcal{PO} = (\mathcal{T}er(\Sigma_\bot, V), \preccurlyeq)$ and $\mathcal{SPO} = (\mathcal{T}er(\Sigma_\bot, V), \prec)$ are respectively a CUSL and a strict partial order.*

Proof. In the case of $\mathcal{PO}$, it follows by induction on the structure of terms that $\preccurlyeq$ is reflexive, transitive, and anti-symmetric. The existence of a least element, the nullary function symbol $\bot$, follows by the second clause of Definition 3.2.1.(1) and anti-symmetry. Moreover, it follows by induction on the structure of terms, again with the help of the second clause of Definition 3.2.1.(1) and anti-symmetry, that each consistent set of terms has a least upper bound. Hence, $\mathcal{PO}$ is a CUSL.

In the case of $\mathcal{SPO}$, it follows by induction of the structure of terms that $\prec$ is transitive and irreflexive. Hence, $\mathcal{SPO}$ is a strict partial order. □

By the above lemma, we have for each $s \in \mathcal{T}er(\Sigma_\bot, V)$ that:

$$\{t \in \mathcal{T}er(\Sigma_\bot, V) \mid t \preccurlyeq s\}$$

is a finite ideal. We already observed this property in the introduction of this section. The ideal is called the *principal ideal* of s.

The following relations hold with regard to positions and the (strict) prefix order:

Proposition 3.2.4. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$.*

1. *For all $s \preccurlyeq t$ it holds that:*
   - *$\mathcal{P}os(s) \subseteq \mathcal{P}os(t)$, and*
   - *$root(s|_p) = root(t|_p)$, if $p \in \mathcal{P}os(s)$ and $s|_p \neq \bot$.*
2. *For all $s \prec t$ there exist $p \in \mathcal{P}os(s)$ such that $s|_p = \bot$ and $t|_p \neq \bot$.*

Proof. By induction on the structure of terms. □

Employing the above proposition, we can prove:

Proposition 3.2.5. *The strict prefix order on $\mathcal{T}er(\Sigma_\perp, V)$ is well-founded.*

Proof. Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$ with $s \prec t$. By Proposition 3.2.4 it follows that:

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

Hence, as $<$ is a well-founded relation on $\mathbb{N}$, the result follows. □

With respect to the prefix order, we have the following property:

Proposition 3.2.6. *Let $S \subseteq \mathcal{T}er(\Sigma_\perp, V)$. The set S has a greatest lower bound with respect to the prefix order.*

Proof. By induction on the structure of terms. □

We next extend the prefix order to substitutions by means of a pointwise definition:

Definition 3.2.7. *Let σ and τ be substitutions. Define:*

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

By induction on the structure of terms it is easily shown that the above definition implies that $\sigma \preccurlyeq \tau$ if and only if $\sigma(s) \preccurlyeq \tau(s)$ for all $s \in \mathcal{T}er(\Sigma_\perp, V)$.

Employing the prefix order on substitutions, we can also extend the strict prefix order to substitutions:

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

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 respectively a partial order and a strict partial order. This follows easily from their respective 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 strict prefix order on substitutions is *not* well-founded. In other words, for each substitution σ there exists a substitution τ such that $\sigma \prec \tau$. To see that this is the case, consider an arbitrary substitution σ and recall that by definition of substitutions there exist $x \in V$ such that $\sigma(x) = x$. If we define $\tau(x) = \perp$ and $\tau(y) = \sigma(y)$ for all $y \neq x$, then $\tau \prec \sigma$.

3.2.2 Infinite Terms

We define the set of infinite terms by means of ideal completion:

Definition 3.2.8. *The set of* infinite terms, *denoted $\mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$, is defined as:*

$$\mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V) = \{I \subseteq \mathcal{T}er(\Sigma_\perp, V) \mid I \text{ is an ideal of } \mathcal{PO}\}\,.$$

Since $\mathcal{PO}$ is a CUSL, it follows that $\mathcal{T}_{\mathrm{i}} = (\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V), \subseteq)$ is a CPO (see Chapter 2). Moreover, it follows for each directed set $D \subseteq \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ that $\bigcup D$ is the least upper bound.

To be notational consistent with (finite) terms, we write $S \preccurlyeq T$ instead of $S \subseteq T$ and $\bigsqcup D$ instead of $\bigcup D$. Moreover, in later chapters we denote $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ by $\mathcal{T}er^{\infty}(\Sigma_{\perp}, V)$ if it is obvious from the context that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ is intended.

Remark 3.2.9. Employing ideals as infinite terms, instead of arbitrary sets of partial terms, has two consequences. First, by employing downward closure and the least upper bound properties of ideals, each (finite) term corresponds to *precisely one* finite element of $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$. Not requiring downward closure would allow both $\{\perp, I\}$ and $\{I\}$ to correspond to I. Moreover, not requiring the least upper bound properties would allow both $\{\perp, I\perp, \perp K, IK\}$ and $\{\perp, I\perp, \perp K\}$ to correspond to KI.

Second, consistency of ideals ensures that *each* finite element of $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ actually corresponds to a (finite) term. Not requiring consistency, would allow us to define the set $\{\perp, I, K\} \subseteq \mathcal{T}er^{\infty}(\Sigma_{\mathrm{CL},\perp}, V)$ and it is unclear to which finite term this set should correspond.

The next characterisation of $S \preccurlyeq T$ is employed numerous times in the following chapters:

Proposition 3.2.10. *For all $S, T \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ it holds that $S \preccurlyeq T$ if and only if for all $s \in S$ there exist $t \in T$ such that $s \preccurlyeq t$.*

Proof. This follows immediately by the fact that $\preccurlyeq$ is subset inclusion and the fact that S and T are downward closed. □

We give an example of an infinite term:

Example 3.2.11. In case of $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\mathrm{CL},\perp}, V)$, we can define the following ideal:

$$I^{\omega} = {\downarrow}\{I^n\perp \mid n \in \mathbb{N}\},$$

where $\downarrow$ denotes downward closure (see Chapter 2) and where $I^n\perp$ is defined as:

$$I^n\perp = \begin{cases} \perp & \text{if } n = 0 \\ I(I^{n-1}\perp) & \text{otherwise} \end{cases}$$

We have for all $n \in \mathbb{N}$ that:

$${\downarrow}\{I^n\} \preccurlyeq I^{\omega}$$

Moreover, we have that:

$$I^{\omega} = \bigsqcup_{n \in \mathbb{N}} {\downarrow}\{I^n\}.$$

The infinite term I^{ω} defined in the above example is identical to the infinite term I^{ω} defined in Example 3.1.2, as we show in the next section. No proof is given here, as we are first required to show that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\mathrm{CL},\perp}, V)$ is a final $F_{\Sigma_{\mathrm{CL},\perp}}$-coalgebra.

With respect to the elements of $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$, we next define the notions of root symbol, positions, subterms, and replacements of subterms. As all elements of $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ are subsets of $\mathcal{T}er(\Sigma_{\perp}, V)$, it is most natural to employ the notions of root symbol, positions, subterms, and replacements of subterms as defined for $\mathcal{T}er(\Sigma_{\perp}, V)$, which is exactly what we do:

Definition 3.2.12. *Let $S, T \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$.*

– *The* root symbol *of S, denoted $root(S)$, is defined as:*

$$root(S) = \begin{cases} \perp & \text{if } S - \{\perp\} = \emptyset \\ root(s) & \text{if } s \in S - \{\perp\} \end{cases}$$

– *The set of* positions *of S, denoted $\mathcal{P}os(S)$, is defined as:*

$$\mathcal{P}os(S) = \bigcup\nolimits_{s \in S} \mathcal{P}os(s)\,.$$

– *If $p \in \mathcal{P}os(S)$, then the* subterm *at position p, denoted $S|_p$ is defined as:*

$$S|_p = \{s|_p \mid s \in S,\ p \in \mathcal{P}os(s)\}\,.$$

– *If $p \in \mathcal{P}os(S)$, then the* replacement *of the subterm at position p by T, denoted $S[T]_p$, is defined as:*

$$S[T]_p = \{s[t]_p \mid s \in S,\ p \in \mathcal{P}os(s),\ t \in T\} \cup \{s \mid s \in S,\ p \notin \mathcal{P}os(s)\}\,.$$

With respect to the above definition, we next establish that *root* is well-defined, that $\mathcal{P}os(S)$ is downward closed, which implies that infinite terms are well-formed, and that $S|_p$ and $S[T]_p$ are again infinite terms:

Proposition 3.2.13. *Let $S, T \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$. The map root is well-defined, the set $\mathcal{P}os(S)$ is downward closed, and if $p \in \mathcal{P}os(S)$, then $S|_p, S[T]_p \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$.*

Proof. That *root* is well-defined follows by Proposition 3.2.4. By the same proposition and the fact that S is an ideal, it follows that $\mathcal{P}os(S)$ is downward closed. That $S|_p$ and $S[T]_p$ are ideals follows immediately by the fact that S and T are ideals. □

Besides the above, it should be immediately obvious that $\mathcal{P}os(S)$, $S|_p$, and $S[T]_p$ are finite in case both S and T are finite. Moreover, in case $root(S) \in \Sigma_n$ we have that:

$$\mathcal{P}os(S) = \{\epsilon\} \cup \{i \cdot p \mid p \in \mathcal{P}os(S|_i) \text{ with } 1 \leqslant i \leqslant n\}$$

and, if $p = i \cdot q$, then we have that:

$$S|_p = (S|_i)|_q$$
$$S[T]_p = (S|_i)[T]_q$$

Finally, the following two properties also hold:

Proposition 3.2.14. *Let $f \in \Sigma_n$ and $S_i \in \mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$ for all $1 \leqslant i \leqslant n$. If the set S is defined as follows:*

$$S = \{\perp\} \cup \{f(s_1, \ldots, s_n) \mid s_i \in S_i\}\,.$$

then $S \in \mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$.

Proof. That S is an ideal follows immediately by the fact that S_i is an ideal for each $1 \leqslant i \leqslant n$. □

Proposition 3.2.15. *Let $S, T \in \mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$. If $root(S) = f = root(T)$ with $f \in \Sigma_n$ and if $S|_i \preccurlyeq T|_i$ for all $1 \leqslant i \leqslant n$, then $S \preccurlyeq T$.*

Proof. If $root(S) = f = root(T)$, then, obviously:

$$S = \{\perp\} \cup \{f(s_1, \ldots, s_n) \mid s_i \in S|_i\}$$

and

$$T = \{\perp\} \cup \{f(t_1, \ldots, t_n) \mid t_i \in T|_i\}\,.$$

Hence, the result follows by the fact that $S|_i \preccurlyeq T|_i$ for all $1 \leqslant i \leqslant n$. □

More insight in the relation between $\mathcal{T}er(\Sigma_\perp, V)$ and $\mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$ can be obtained by defining a map, denoted ι, from $\mathcal{T}er(\Sigma_\perp, V)$ to $\mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$ that assigns to each term its principal ideal. That is, ι is defined as:

$$\iota(s) = {\downarrow}\{s\} = \{t \in \mathcal{T}er(\Sigma_\perp, V) \mid t \preccurlyeq s\}\,,$$

where $s \in \mathcal{T}er(\Sigma_\perp, V)$.

It is readily shown that ι is an isomorphism between $\mathcal{T}er(\Sigma_\perp, V)$ and the finite ideals of $\mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$. The inverse of ι assigns to each finite ideal $I \in \mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$ its least upper bound:

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

The existence of the least upper bound of I is immediate by the fact that each finite ideal has a maximal element.

In case s and t are terms in $\mathcal{T}er(\Sigma_\perp, V)$ and S and T are finite ideals of $\mathcal{T}er^\infty_{\mathrm{i}}(\Sigma_\perp, V)$, a number of properties hold:

$$\begin{aligned}
s \preccurlyeq t &\Leftrightarrow \iota(s) \preccurlyeq \iota(t) & \qquad S \preccurlyeq T &\Leftrightarrow \iota^{-1}(S) \preccurlyeq \iota^{-1}(T)\\
root(s) &= root(\iota(s)) & root(S) &= root(\iota^{-1}(S))\\
\mathcal{P}os(s) &= \mathcal{P}os(\iota(s)) & \mathcal{P}os(S) &= \mathcal{P}os(\iota^{-1}(S))\\
\iota(s|_p) &= \iota(s)|_p & \iota^{-1}(S|_p) &= \iota^{-1}(S)|_p\\
\iota(s[t]_p) &= \iota(s)[\iota(t)]_p & \iota^{-1}(S[T]_p) &= \iota^{-1}(S)[\iota^{-1}(T)]_p
\end{aligned}$$

Employing Proposition 3.2.10, all properties follow easily from their respective definitions.

3.2.3 Final Coalgebra

In this section, we show that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ is a final $F_{\Sigma_{\bot}}$-coalgebra. We concern ourselves with $F_{\Sigma_{\bot}}$-coalgebras instead F_{Σ}-coalgebras, as we employ $F_{\Sigma_{\bot}}$-coalgebras, and not F_{Σ}-coalgebras, in Chapters 5 through 8.

Recall that any $F_{\Sigma_{\bot}}$-coalgebra is a pair (A, α), with A a set and α a total map from A to $F_{\Sigma_{\bot}}(A)$. Hence, we cannot actually show that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ is a final $F_{\Sigma_{\bot}}$-coalgebra. Instead, what we need to do, is to define a map from $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ to $F_{\Sigma_{\bot}}(\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V))$ and to show that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ together with the map defines a final $F_{\Sigma_{\bot}}$-coalgebra.

We define the map from $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ to $F_{\Sigma_{\bot}}(\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V))$, denoted φ_{i}, as follows, supposing $S \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$:

$$\varphi_{\mathrm{i}}(S) = \begin{cases} x & \text{if } root(S) = x \in V \\ (f, S|_1, \ldots, S|_n) & \text{if } root(S) = f \in \Sigma_n \end{cases}$$

The definition agrees with the informal description of φ_{∞} in Section 3.1.1: If we apply φ_{i} to an infinite term which is not a variable, then we obtain the root symbol $f \in \Sigma_n$ and its n subterms. Otherwise, if we apply φ_{i} to a variable, then we obtain that variable.

The above explanation of φ_{i} also clarifies why we do not show that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ is a final F_{Σ}-coalgebra: There are no maps from $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ to $F_{\Sigma}(\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V))$ that agree with the informal description of φ_{∞}. To understand this, observe that $\bot \notin \Sigma$ implies $\bot \notin F_{\Sigma}(\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V))$. As such, it is impossible to define a map from $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ to $F_{\Sigma}(\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V))$ which allows us to obtain $\bot$ whenever we apply the map to the infinite term $\{\bot\}$. Hence, any map from $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ to $F_{\Sigma}(\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V))$ will disagree with the informal description of φ_{∞}.

Remark 3.2.16. It is easy to define a subset of $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ for which there *does* exist a map that agrees with the informal description of φ_{∞}. Denoting the subset by $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma, V)$, we define:

$$\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma, V) = \{S \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V) \mid \forall p \in \mathcal{P}os(S) : S|_p \neq \{\bot\}\}.$$

As follows immediately by definition of φ_{i}, the informal description of φ_{∞} is satisfied by restriction of φ_{i} to $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma, V)$.

The set $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma, V)$ and the restriction of φ_{i} define a final F_{Σ}-coalgebra. To prove this, an argument is required that is very similar to the one below showing that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ and φ_{i} define a final $F_{\Sigma_{\bot}}$-coalgebra.

Now define the following pair:

$$(\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V), \varphi_{\mathrm{i}})$$

We next prove that the pair forms a final $F_{\Sigma_{\bot}}$-coalgebra. To facilitate the proof, we define for each $S \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ the *restriction* of S to depth $n \in \mathbb{N}$, denoted $S{\restriction}_n$:

$$S{\restriction}_n = \{s \in S \mid \forall p \in \mathcal{P}os(s) : |p| \leqslant n\}.$$

With respect to restrictions, we have:

Proposition 3.2.17. *If $S \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ and $n \in \mathbb{N}$, then it holds that $S{\restriction}_n \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$.*

Proof. That $\bot$ is an element of $S{\restriction}_n$ is immediate by $\mathcal{P}os(\bot) = \{\epsilon\}$ and $|\epsilon| = 0$. Consistency and the existence of a least upper bound for each $\{s, t\} \subseteq S{\restriction}_n$ follows by the fact that $\{s, t\} \subseteq S$ and by the fact that S is an ideal. Finally, that $s \sqcup t \in S{\restriction}_n$ follows by observing that $p \in \mathcal{P}os(s \sqcup t)$ implies either $p \in \mathcal{P}os(s)$ or $p \in \mathcal{P}os(t)$. □

To prove that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$ is a final $F_{\Sigma_{\bot}}$-coalgebra, we first prove for infinite terms that bisimilarity (see Definition 3.1.3) implies equality. That is, we first prove that $S \sim T$ implies $S = T$. Remark that the reverse, i.e., $S = T$ implies $S \sim T$, is immediate by definition of equality.

Lemma 3.2.18. *Let $S, T \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$. If $S \sim T$, then $S = T$.*

Proof. Suppose $S \sim T$. To start, we prove by induction on $n \in \mathbb{N}$ that $S{\restriction}_n \preccurlyeq T$ for all n.

Base Case. Let $n = 0$. The only possibilities for $S{\restriction}_0$ are $\{\bot\}$, $\{\bot, x\}$ and $\{\bot, c\}$ with $x \in V$ and $c \in \Sigma_0$.

If $S{\restriction}_0 = \{\bot\}$, then we immediately have $S{\restriction}_0 \preccurlyeq T$ since T is an ideal. Otherwise, if $S{\restriction}_0 = \{\bot, t\}$ with either $t = x$ or $t = c$, then we have $\varphi_{\mathrm{i}}(S{\restriction}_0) = t$ and, by definition of ideals, $S = S{\restriction}_0$. Hence, as $S \sim T$, it follows by definition of bisimulation for F_{Σ}-coalgebras that $\varphi_{\mathrm{i}}(T) = t$ and $T = \{\bot, t\}$. Thus, $S{\restriction}_0 \preccurlyeq T$.

Induction Step. Suppose the result holds for some $n \geqslant 0$. We prove the result for $n + 1$. As $n + 1 > 0$, we have either $S{\restriction}_{n+1} = S{\restriction}_n$ or $S{\restriction}_{n+1} \neq S{\restriction}_n$. In the first case, $S{\restriction}_{n+1} \preccurlyeq T$ is immediate by the induction hypothesis.

In the second case, we have that $root(S{\restriction}_{n+1}) = root(S) = f$ for some $f \in \Sigma_m$ and $m > 0$. Otherwise, the first case holds. Since $S \sim T$, we have by definition of bisimulation that $root(T) = f$ and that $S|_i \sim T|_i$ for all $1 \leqslant i \leqslant m$. Moreover, by the induction hypothesis, $(S{\restriction}_{n+1})|_i = (S|_i){\restriction}_n \preccurlyeq T|_i$ for all $1 \leqslant i \leqslant m$. Hence, we have by Proposition 3.2.15 that $S{\restriction}_{n+1} \preccurlyeq T$.

As we have for all $s \in S$ that there exists an $m \in \mathbb{N}$ such that $|p| \leqslant m$ for all $p \in \mathcal{P}os(s)$, it follows that $S = \bigsqcup_{n \in \mathbb{N}} S{\restriction}_n$. Hence, $S \sim T$ implies $S \preccurlyeq T$. Since $S \sim T$ implies $T \sim S$, it also follows that $T \preccurlyeq S$ and $S = T$, as required. □

Employing the above lemma, we next prove the main theorem of this section:

Theorem 3.2.19. *The pair $(\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V), \varphi_{\mathrm{i}})$ is a final $F_{\Sigma_{\bot}}$-coalgebra.*

Proof. Let (A, α) be a $F_{\Sigma_{\bot}}$-coalgebra. We define a map from A to $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\bot}, V)$, denoted t_A^n, supposing $a \in A$ and $n \in \mathbb{N}$:

$$t_A^0(a) = \{\bot\}$$

$$t_A^{n+1}(a) = \begin{cases} \{\bot, x\} & \text{if } \alpha(a) = x \text{ with } x \in V \\ \{\bot\} \cup \{f(s_1, \ldots, s_m) \mid s_i \in S_i\} & \text{if } \pi_1 \circ \alpha(a) = f \text{ with } f \in \Sigma_m \\ & \quad \text{and } S_i = t_A^n(\pi_{i+1} \circ \alpha(a)) \end{cases}$$

By induction on n, employing Propositions 3.2.14 and 3.2.15, it follows that $t_A^n(a)$ is an ideal and that $t_A^n(a) \preccurlyeq t_A^{n+1}(a)$. Hence, as $\mathcal{T}_{\mathrm{i}} = (\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V), \preccurlyeq)$ is a CPO, we have that the ideal $\bigsqcup\{t_A^n(a) \mid n \in \mathbb{N}\}$ exists and we can define:

$$t_A(a) = \bigsqcup\{t_A^n(a) \mid n \in \mathbb{N}\}\,.$$

By definition of φ_{i} and t_A^n it follows easily that t_A is an $F_{\Sigma_{\perp}}$-homomorphism. Hence, there exists a homomorphism from each $F_{\Sigma_{\perp}}$-coalgebra to $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$.

To show that the homomorphism is unique, suppose for some coalgebra (A, α) that there exist two homomorphisms from A to $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$, denote them f and g. Define the relation $R = \{(f(a), g(a)) \mid a \in A\}$. As noted in Chapter 2, the relation R is a bisimulation. Because $R \subseteq \sim$, we have by Lemma 3.2.18 that $f(a) = g(a)$ for all $a \in A$. Hence, $f = g$ and the homomorphism t_A is unique. □

The above method of proving that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ is a final $F_{\Sigma_{\perp}}$-coalgebra is fairly standard: Rutten [Rut98], e.g., gives an almost identical proof to show that $\wp(A^*)$ is a final D_A-coalgebra.

Example 3.2.20. Let m be the map from $\mathbf{1}$ to $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\mathrm{CL},\perp}, V)$ such that we obtain I^{ω}, as defined in Example 3.2.11, whenever we apply m to $*$. The map m is the unique homomorphism that makes the diagram in Example 3.1.2 commute, assuming that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\mathrm{CL},\perp}, V)$ is the employed final coalgebra. That m makes the diagram commute is immediate by its definition. Uniqueness of m follows by the fact that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\mathrm{CL},\perp}, V)$ is a final $F_{\Sigma_{\mathrm{CL},\perp}}$-coalgebra.

By the properties of m, we have that the infinite term I^{ω}, as defined in Example 3.1.2, is identical to I^{ω}, as defined in Example 3.2.11, in case $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\mathrm{CL},\perp}, V)$ is employed as final coalgebra, as already hinted at just below Example 3.2.11.

3.2.4 Homomorphisms

We next show for $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ that the usual definitions of positions, subterms, and replacements of subterms are identical to the definitions given in Sections 3.1.2, 3.1.3, and 3.1.4. The reasoning employed in the proofs is very similar to that of the example in the previous section.

Positions. To prove that the definition of $\mathcal{P}os$ for $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ corresponds to $\mathcal{P}os$ as defined in Section 3.1.2, we define a map $\mathcal{P}os'$ from $\mathbf{1} + \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ to $\wp(\mathbb{N}^*)$:

$$\mathcal{P}os'(x) = \begin{cases} \emptyset & \text{if } x \in \mathbf{1} \\ \mathcal{P}os(x) & \text{if } x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V) \end{cases}$$

Obviously, $\mathcal{P}os(S) = \mathcal{P}os' \circ \mathrm{inr}(S)$ for $S \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$. Hence, we only need to prove:

Lemma 3.2.21. *Let $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ be employed as final coalgebra. The map:*

$$\mathcal{P}os' : \mathbf{1} + \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V) \to \wp(\mathbb{N}^*)$$

is the unique homomorphism defined in Section 3.1.2.

Proof. To prove that $\mathcal{P}os'$ is a homomorphism we have to show for all $x \in \mathbf{1} + \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ that $D_{\mathbb{N}}(\mathcal{P}os') \circ \langle o_{\mathrm{p}}, t_{\mathrm{p}} \rangle(x) = \langle \mathrm{o}_{\mathbb{N}}, \mathrm{t}_{\mathbb{N}} \rangle \circ \mathcal{P}os'(x)$. There are two cases to consider, depending on x: Either $x \in \mathbf{1}$ or $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$.

In case $x \in \mathbf{1}$, we have $x = *$, $o_{\mathrm{p}}(*) = 0$, $t_{\mathrm{p}}(*)(i) = *$, and $\mathcal{P}os'(*) = \emptyset$. Hence, for all $i \in \mathbb{N}$:

$$1_{\mathbf{2}} \circ o_{\mathrm{p}}(*) = 0 = \mathrm{o}_{\mathbb{N}} \circ \mathcal{P}os'(*)$$
$$\mathcal{P}os' \circ t_{\mathrm{p}}(*)(i) = \emptyset = \mathrm{t}_{\mathbb{N}} \circ \mathcal{P}os'(*)(i)$$

This implies $D_{\mathbb{N}}(\mathcal{P}os') \circ \langle o_{\mathrm{p}}, t_{\mathrm{p}} \rangle(x) = \langle \mathrm{o}_{\mathbb{N}}, \mathrm{t}_{\mathbb{N}} \rangle \circ \mathcal{P}os'(x)$ whenever $x = *$.

In case $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$, assume $x = S$. There are two possibilities: Either $root(S) \in V$ or $root(S) \in \Sigma_n$. However, in both cases we have that $\epsilon \in \mathcal{P}os'(S) = \mathcal{P}os(S)$, which implies:

$$1_{\mathbf{2}} \circ o_{\mathrm{p}}(S) = 1 = \mathrm{o}_{\mathbb{N}} \circ \mathcal{P}os'(S)\,.$$

If we have $root(S) \in V$ or $i > n$, then $t_{\mathrm{p}}(S)(i) = *$ and $\mathcal{P}os(S)|_i = \emptyset$, which gives:

$$\mathcal{P}os' \circ t_{\mathrm{p}}(S)(i) = \emptyset = \mathrm{t}_{\mathbb{N}} \circ \mathcal{P}os'(S)(i)\,.$$

Otherwise, if $root(S) \in \Sigma_n$ and $1 \leqslant i \leqslant n$, then $t_{\mathrm{p}}(S)(i) = S|_i$, which gives:

$$\mathcal{P}os' \circ t_{\mathrm{p}}(S)(i) = \mathcal{P}os(S|_i) = \mathrm{t}_{\mathbb{N}} \circ \mathcal{P}os'(S)(i)\,,$$

where we employ the fact that $\mathcal{P}os'(S|_i) = \mathcal{P}os(S|_i) = \mathcal{P}os'(S)|_i$. Thus, $x = S$ implies $D_{\mathbb{N}}(\mathcal{P}os') \circ \langle o_{\mathrm{p}}, t_{\mathrm{p}} \rangle(x) = \langle \mathrm{o}_{\mathbb{N}}, \mathrm{t}_{\mathbb{N}} \rangle \circ \mathcal{P}os'(x)$. Combining this result with the result for $x = *$, we obtain that $\mathcal{P}os'$ is a homomorphism. Uniqueness of the homomorphism follows by the fact that $\wp(\mathbb{N}^*)$ is a final coalgebra. □

Subterms. To prove that the definition of $S|_p$ for $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ corresponds to $S|_p$ as defined in Section 3.1.3, we proceed in a similar fashion as in the case of $\mathcal{P}os$. We define a map $\_|'_\_$ from $\mathcal{TP}$ to $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V) \times \mathcal{TP}$:

$$\_|'_\_(S, p) = (S|_p, (S, p))\,.$$

Obviously, $S|_p = \pi_1 \circ \_|'_\_(S, p)$ for $(S, p) \in \mathcal{TP}$ and we only need the following:

Lemma 3.2.22. *Let $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ be employed as final coalgebra. The map:*

$$\_|'_\_ : \mathcal{TP} \to \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V) \times \mathcal{TP}$$

is the unique homomorphism defined in Section 3.1.3.

Proof. To prove that $\_|'_\_$ is a homomorphism we have to show for all $x \in TP(\mathcal{TP})$ that $\_|'_\_ \circ \psi_{\mathrm{tp}}(x) = \psi_{\mathrm{s}} \circ TP(\_|'_\_)(x)$. There are two cases to consider, depending on x: Either $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$ or $x \in TP_i^n(\mathcal{TP})$.

In case $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$, assume $x = S$. We have $\psi_{\mathrm{tp}}(S) = (S, \epsilon)$ and $TP(\_|'_\_)(S) = S$, which gives:

$$\_|'_\_ \circ \psi_{\mathrm{tp}}(S) = (S, (S, \epsilon)) = \psi_{\mathrm{s}} \circ TP(\_|'_\_)(S)\,.$$

Hence, $\_|'_\_ \circ \psi_{\mathrm{tp}}(x) = \psi_{\mathrm{s}} \circ TP(\_|'_\_)(x)$ whenever $x = S$ with $S \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_{\perp}, V)$.

In case $x \in TP_i^n(\mathcal{TP})$, assume $x = (f, y_1, \ldots, y_n)$ with $y_i = (S_i, p)$ and $y_j = S_j$ for all $j \neq i$. We have $\psi_{\mathrm{tp}}(x) = (S, i \cdot p)$ with $S = \{f(s_1, \ldots, s_n) \mid s_k \in S_k \text{ for } 1 \leqslant k \leqslant n\}$, and $TP(\_|'_\_)(x) = (f, y'_1, \ldots, y'_n)$ with $y'_i = (S_i|_p, (S_i, p))$ and $y'_j = S_j$ for all $j \neq i$. This gives:

$$\_|'_\_ \circ \psi_{\mathrm{tp}}(x) = (S_i|_p, (S, i \cdot p)) = \psi_{\mathrm{s}} \circ TP(\_|'_\_)(x)\,,$$

where we employ that $S|_{i \cdot p} = S_i|_p$. Thus, $x \in TP_i^n(\mathcal{TP})$ implies $\_|'_\_ \circ \psi_{\mathrm{tp}}(x) = \psi_{\mathrm{s}} \circ TP(\_|'_\_)(x)$. Combining this result with the result for $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V)$ implies that $\_|'_\_$ is a homomorphism. Uniqueness of the homomorphism follows by the fact that $\mathcal{TP}$ is an initial algebra. □

Replacements of Subterms. As before, we prove that the definition of $S[T]_p$ for $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V)$ corresponds to $S[T]_p$ as defined in Section 3.1.4 by defining another map. In this case, we define a map $\_[\_]'_\_$ from $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V) + \mathcal{TPT}$ to $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V)$:

$$\_[\_]'_\_(x) \begin{cases} x & \text{if } x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V) \\ S[T]_p & \text{if } x = (S, p, T) \in \mathcal{TPT} \end{cases}$$

Obviously, $S[T]_p = \_[\_]'_\_ \circ \mathrm{inr}(S, p, T)$ for all $(S, p, T) \in \mathcal{TPT}$. Hence, we only need to prove:

Lemma 3.2.23. *Let $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V)$ be employed as final coalgebra. The map:*

$$\_[\_]'_\_ : \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V) + \mathcal{TPT} \to \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V)$$

is the unique homomorphism defined in Section 3.1.4.

Proof. To prove that $\_[\_]'_\_$ is a homomorphism we have to show that $F_{\Sigma_\bot}(\_[\_]'_\_) \circ \varphi_{\mathrm{r}}(x) = \varphi_{\mathrm{i}} \circ \_[\_]'_\_(x)$ for all $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V) + \mathcal{TPT}$. There are two cases to consider, depending on x: Either $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V)$ or $x \in \mathcal{TPT}$.

In case $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V)$, assume $x = S$. We have $\varphi_{\mathrm{r}}(S) = F_{\Sigma_\bot}(\mathrm{inl}) \circ \varphi_{\mathrm{i}}(S)$ and $\_[\_]'_\_(S) = S$, which gives:

$$F_{\Sigma_\bot}(\_[\_]'_\_) \circ \varphi_{\mathrm{r}}(S) = \varphi_{\mathrm{i}}(S) = \varphi_{\mathrm{i}} \circ \_[\_]'_\_(S)\,.$$

Hence, $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\bot, V)$ implies $F_{\Sigma_\bot}(\_[\_]'_\_) \circ \varphi_{\mathrm{r}}(x) = \varphi_{\mathrm{i}} \circ \_[\_]'_\_(x)$.

In case $x \in \mathcal{TPT}$, assume $x = (S, p, T)$. There are two cases to consider, depending on p: Either $p = \epsilon$ or $p = i \cdot q$. In the first case, we have $\varphi_{\mathrm{r}}(S, p, T) = F_{\Sigma_\bot}(\mathrm{inl}) \circ \varphi_{\mathrm{i}}(T)$ and $\_[\_]'_\_(S, p, T) = T$, which gives:

$$F_{\Sigma_\bot}(\_[\_]'_\_) \circ \varphi_{\mathrm{r}}(S, p, T) = \varphi_{\mathrm{i}}(T) = \varphi_{\mathrm{i}} \circ \_[\_]'_\_(S, p, T)\,.$$

In the second case, we have $\varphi_{\mathrm{r}}(S, p, T) = (f, y_1, \ldots, y_n)$ with $root(S) = f$, $f \in \Sigma_n$, $y_i = (S|_i, q, T)$, and $y_j = S|_j$ for all $j \neq i$. Moreover, we have $\_[\_]'_\_(S, p, T) = S[T]_p$. This gives:

$$F_{\Sigma_\bot}(\_[\_]'_\_) \circ \varphi_{\mathrm{r}}(S, p, T) = (f, y'_1, \ldots, y'_n) = \varphi_{\mathrm{i}} \circ \_[\_]'_\_(S, p, T)\,,$$

where $y'_i = (S|_i)[T]_q$ and $y'_j = y_j$ for all $j \neq i$. Here, we employ the fact that $(S[T]_{i\cdot q})|_i = (S|_i)[T]_q$. Thus, $x \in \mathcal{TPT}$ implies $F_{\Sigma_\perp}(\_[\_]'_\_) \circ \varphi_{\mathrm{r}}(x) = \varphi_{\mathrm{i}} \circ \_[\_]'_\_(x)$. Combining this result with the result for $x \in \mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\perp, V)$ implies that $\_[\_]'_\_$ is a homomorphism. Uniqueness of the homomorphism follows by the fact that $\mathcal{T}er_{\mathrm{i}}^{\infty}(\Sigma_\perp, V)$ is a final coalgebra. □

3.3 Partial Functions

Besides representing a (finite) term by the set of all its prefixes, we can also represent it as a partial function s from $\mathbb{N}^*$ to $\Sigma \cup V$. The partial function satisfies a number of properties:

1. the set of values on which s is defined is downward closed and finite,
2. if $s(p) \in V$, then $s(q)$ is undefined for all $q > p$, and
3. if $s(p) \in \Sigma_n$, then $s(p \cdot i)$ is defined for all $1 \leqslant i \leqslant n$ and undefined otherwise.

Infinite terms are defined by dropping the finiteness restriction from the first clause. As such, the approach is very similar to ideal completion.

The definition of infinite terms by means of partial functions is the subject of Section 3.3.1. In Sections 3.3.2 and 3.3.3 we prove respectively that the partial functions define a final F_Σ-coalgebra and that the morphisms as defined in Section 3.1 correspond to the ones defined for partial functions.

Bibliographic Notes. Infinite terms defined by means of partial functions are discussed in the seminal paper by Goguen, Thatcher, Wagner, and Wright [GTWW77], among other publications. They also figure in the papers on recursive program schemes by Courcelle and Nivat [CN78] and Courcelle [Cou79]. A fundamental study of infinite terms defined by means of partial functions occurs in a paper by Courcelle [Cou83].

3.3.1 Infinite Terms

We define the set of infinite terms by means of partial functions:

Definition 3.3.1. *The set $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$ is the set of all partial functions $S : \mathbb{N}^* \rightharpoonup (\Sigma \cup V)$ satisfying:*

1. *the set of values on which S is defined is downward closed,*
2. *if $S(p) \in V$, then $S(q)$ is undefined for all $q > p$, and*
3. *if $S(p) \in \Sigma_n$, then $S(p \cdot i)$ is defined for all $1 \leqslant i \leqslant n$ and undefined otherwise,*

where $p \in \mathbb{N}^*$.

Obviously, the elements of $\mathcal{T}er(\Sigma, V)$ form a subset of $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$, as those elements simply have a finiteness restriction added to the first clause.

Remark 3.3.2. The first clause of the above definition is similar to the requirement that each ideal is downward closed (see Remark 3.2.9). The other clauses ensure that the restrictions imposed by the signature are satisfied.

We give an example of an infinite term:

Example 3.3.3. In case of Σ_{CL}, define I^ω as the following partial function:

$$I^\omega(p) = \begin{cases} \cdot & \text{if } p = 2^n \\ I & \text{if } p = 2^n \cdot 1 \\ \text{undefined} & \text{otherwise} \end{cases}$$

That I^ω is an element of $\mathcal{T}er_{\mathrm{f}}^\infty(\Sigma_{\mathrm{CL}}, V)$ is easy to see: It is downward closed, because $2^0 = \epsilon$, $2^{n-1} < 2^n$, and $2^n < 2^n \cdot 1$. Moreover, the other requirements are satisfied, because $ar(\cdot) = 2$ and $ar(I) = 0$.

After proving that $\mathcal{T}er_{\mathrm{f}}^\infty(\Sigma, V)$ is a final F_Σ-coalgebra, we could show that I^ω, as defined above, is identical to the infinite term I^ω defined in Example 3.1.2, analogous to what we did in the case of Example 3.2.11. However, we omit such a proof, as it is identical to the proof in Example 3.2.20.

We next define the notions of root symbol, positions, subterms, and replacements of subterms for the elements of $\mathcal{T}er_{\mathrm{f}}^\infty(\Sigma, V)$:

Definition 3.3.4. *Let $S, T \in \mathcal{T}er_{\mathrm{f}}^\infty(\Sigma, V)$.*

- *The* root symbol *of S, denoted $root(S)$, is defined as:*

$$root(S) = S(\epsilon)\,.$$

- *The set of* positions *of S, denoted $\mathcal{P}os(S)$, is defined as:*

$$\mathcal{P}os(S) = \{p \in \mathbb{N}^* \mid S(p) \text{ is defined}\}\,.$$

- *If $p \in \mathcal{P}os(S)$, then the* subterm *at position p, denoted $S|_p$ is defined, for all $q \in \mathbb{N}^*$, as:*

$$S|_p(q) = S(p \cdot q)\,.$$

- *If $p \in \mathcal{P}os(S)$, then the* replacement *of the subterm at position p by T, denoted $S[T]_p$, is defined, for all $q \in \mathbb{N}^*$, as:*

$$S[T]_p(q) = \begin{cases} S(q) & \text{if either } q \parallel p \text{ or } q < p \\ T(r) & \text{if } p = q \cdot r \end{cases}$$

We have the following:

Proposition 3.3.5. *Let $S, T \in \mathcal{T}er_{\mathrm{f}}^\infty(\Sigma, V)$. The map root is well-defined, the set $\mathcal{P}os(S)$ is downward closed, and if $p \in \mathcal{P}os(S)$, then $S|_p, S[T]_p \in \mathcal{T}er_{\mathrm{f}}^\infty(\Sigma, V)$.*

Proof. Immediate by the definitions. □

It should be directly obvious that $\mathcal{P}os(S)$, $S|_p$, and $S[T]_p$ are finite in case S and T are finite. Moreover, in case $root(S) \in \Sigma_n$ we have that:

$$\mathcal{P}os(S) = \{\epsilon\} \cup \{i \cdot p \mid p \in \mathcal{P}os(S|_i) \text{ and } 1 \leqslant i \leqslant n\}$$

and in case $p = i \cdot q$ we have that:

$$S|_p = (S|_i)|_q$$
$$S[T]_p = (S|_i)[T]_q$$

3.3.2 Final Coalgebra

We next show that $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$ is a final F_Σ-coalgebra. We proceed in a similar fashion as in Section 3.2.3. That is, we prove that $S \sim T$ implies $S = T$ before proving that $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$ is a final F_Σ-coalgebra.

Of course, we first need to define a map from $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$ to $F_\Sigma(\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V))$. The map, denoted φ_{f}, is defined as follows, for all $S \in \mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$:

$$\varphi_{\mathrm{f}}(S) = \begin{cases} x & \text{if } root(S) = x \in V \\ (f, S|_1, \ldots, S|_n) & \text{if } root(S) = f \in \Sigma_n \end{cases}$$

As is easy to see, the map adheres to the informal description of φ_∞ as presented in Section 3.1.1, just like φ_{i} in Section 3.2.3.

We next show that

$$(\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V), \varphi_{\mathrm{f}})$$

is a final F_Σ-coalgebra, following the steps mentioned above.

Lemma 3.3.6. *Let $S, T \in \mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$. If $S \sim T$, then $S = T$.*

Proof. Suppose $S \sim T$. We prove for all $p \in \mathbb{N}^*$ that $S(p)$ and $T(p)$ are both undefined or that $S(p) = T(p)$. The proof is by induction on $|p|$. That $S = T$ then follows, as all $p \in \mathbb{N}^*$ have finite length.

Base Case. In this case, $p = \epsilon$. By definition of $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$ we have that $S(\epsilon)$ and $T(\epsilon)$ are both defined. That $S(\epsilon) = T(\epsilon)$ is immediate by $S \sim T$.

Induction Step. Suppose the result holds for some $|q| \geqslant 0$. We prove the result for $p = q \cdot i$. By the induction hypothesis, there are two cases to consider: Either $S(q)$ and $T(q)$ are both undefined or $S(q) = T(q)$.

In case $S(q)$ and $T(q)$ are both undefined, we have by definition of $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$ that both $S(p)$ and $T(p)$ are undefined. Hence, the result follows.

In case $S(q) = T(q)$, there are two possibilities: Either $S(q)$ and $T(q)$ are identical variables or they are identical function symbols. In case $S(q)$ and $T(q)$ are identical variables, the result is immediate, as $S(p)$ and $T(p)$ are then both undefined by definition $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$. In the other case, depending on the value of i, we have that $S(p)$ and $T(p)$ are either both defined or both undefined. Hence, the result follows if we can prove $S(p) = T(p)$ whenever $S(p)$ and $T(p)$ are both defined, which is immediate by $S \sim T$. □

We next prove the main theorem of this section:

Theorem 3.3.7. *The pair $(\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V), \varphi_{\mathrm{f}})$ is a final F_Σ-coalgebra.*

Proof. Let (A, α) be a F_Σ-coalgebra. We define a map from A to $\mathcal{T}er_{\mathrm{f}}^{\infty}(\Sigma, V)$, denoted t_A, given $a \in A$ and $p \in \mathbb{N}^*$:

$$t_A(a)(\epsilon) = \begin{cases} x & \text{if } \alpha(a) = x \text{ with } x \in V \\ f & \text{if } \pi_1 \circ \alpha(a) = f \text{ with } f \in \Sigma \end{cases}$$

$$t_A(a)(i \cdot p) = \begin{cases} t_A(\pi_{i+1} \circ \alpha(a))(p) & \text{if } \pi_1 \circ \alpha(a) \in \Sigma_n \text{ and } 1 \leqslant i \leqslant n \\ \text{undefined} & \text{otherwise} \end{cases}$$

By induction on the length of $p \in \mathbb{N}^*$ it follows that $t_A(a) \in \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$ for all $a \in A$. Moreover, by definition of φ_{f} it follows easily that t_A is a homomorphism. Hence, there exists a homomorphism from each F_Σ-coalgebra to $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$.

To show that the homomorphism is unique, suppose that for some coalgebra (A, α) there exist two homomorphisms from A to $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$, denote them f and g. Define the relation $R = \{(f(a), g(a)) \mid a \in A\}$. As noted in Chapter 2, the relation R is a bisimulation. Because $R \subseteq \sim$, we have by Lemma 3.3.6 that $f(a) = g(a)$ for all $a \in A$. Hence, $f = g$ and the homomorphism t_A is unique. □

3.3.3 Homomorphisms

We next prove that the definitions of positions, subterms, and replacements of subterms as given for $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$ are identical to the definitions given in Sections 3.1.2, 3.1.3, and 3.1.4 in case $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$ is employed as final F_Σ-coalgebra.

Positions. Define a map $\mathcal{P}os'$ from $\mathbf{1} + \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$ to $\wp(\mathbb{N}^*)$:

$$\mathcal{P}os'(x) = \begin{cases} \emptyset & \text{if } x \in \mathbf{1} \\ \mathcal{P}os(x) & \text{if } x \in \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V) \end{cases}$$

Obviously, $\mathcal{P}os(S) = \mathcal{P}os' \circ \mathrm{inr}(S)$ for $S \in \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$. Hence, we only need to prove:

Lemma 3.3.8. *Let $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$ be employed as final coalgebra. The map:*

$$\mathcal{P}os' : \mathbf{1} + \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V) \to \wp(\mathbb{N}^*)$$

is the unique homomorphism defined in Section 3.1.2.

Proof. Completely analogous to the proof of Lemma 3.2.21. □

Subterms. Define a map $\_|'\_$ from $\mathcal{TP}$ to $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V) \times \mathcal{TP}$:

$$\_|'\_(S, p) = (S|_p, (S, p)) .$$

Obviously, $S|_p = \pi_1 \circ \_|'\_(S, p)$ for $(S, p) \in \mathcal{TP}$ and we only need to prove:

Lemma 3.3.9. *Let $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$ be employed as final coalgebra. The map:*

$$\_|'\_ : \mathcal{TP} \to \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V) \times \mathcal{TP}$$

is the unique homomorphism defined in Section 3.1.3.

Proof. Completely analogous to the proof of Lemma 3.2.22. □

Replacements of Subterms. Define a map $\_[\_]'\_$ from $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V) + \mathcal{TPT}$ to $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V)$:

$$\_[\_]'\_(x) \begin{cases} x & \text{if } x \in \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma, V) \\ S[T]_p & \text{if } x = (S, p, T) \in \mathcal{TPT} \end{cases}$$

Obviously, $S[T]_p = \_[\_]'_\_ \circ \mathrm{inr}(S,p,T)$ for all $(S,p,T) \in \mathcal{TPT}$. Hence, we only need to prove:

Lemma 3.3.10. *Let $\mathcal{T}er^\infty_{\mathrm{f}}(\Sigma,V)$ be employed as final coalgebra. The map:*

$$\_[\_]'_\_ : \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma,V) + \mathcal{TPT} \to \mathcal{T}er^\infty_{\mathrm{f}}(\Sigma,V)$$

is the unique homomorphism defined in Section 3.1.4.

Proof. Completely analogous to the proof of Lemma 3.2.23. □

3.4 Metric Completion

The definition of infinite terms by means of metric completion is based on the observation that two terms s and t are in some sense 'close' to each other when all positions $p \in \mathcal{P}os(s) \cap \mathcal{P}os(t)$ with $root(s|_p) \neq root(t|_p)$ occur at great depth. The observation allows for the introduction of a metric. Hence, it becomes possible to define Cauchy sequences of terms, which in turn allows for metric completion. The infinite terms are defined as the elements of the space such obtained.

In Section 3.4.1, we define the metric spaces employed in the definitions of infinite terms, root symbols, positions, subterms, and replacements of subterms. Among the metric spaces is the metric space informally described above. In Sections 3.4.2 and 3.4.3, we respectively define infinite terms by means of metric completion and we prove that the definition yields a final F_Σ-coalgebra. Finally, in Section 3.4.4, we prove that the morphisms defined in Section 3.1 correspond to the ones defined in the case of metric completion.

Bibliographic Notes. Defining infinite terms by means of metric completion finds its origins in the work by Arnold and Nivat [AN80]. The definition is often employed in infinitary rewriting, see, e.g., Chapter 12 in the book by Terese [Ter03] and the papers on infinitary rewriting by Dershowitz, Kaplan, and Plaisted [DKP91] and Kennaway, Klop, Sleep, and De Vries [KKSV95].

3.4.1 Metric Spaces

The metric spaces employed in Section 3.4.2 are summarised the table below. The spaces that occur in the table but that have not yet been defined are defined next.

Space	Remark
$(\Sigma \cup V, d_{\mathrm{d}})$	d_{d} is the discrete metric
$(\mathbb{N}^*, d_{\mathrm{B}})$	d_{B} is the Baire metric
$(\wp_{\mathrm{nc}}(\mathbb{N}^*), d^+_{\mathrm{B}})$	$\wp_{\mathrm{nc}}(\mathbb{N}^*)$ is the set of all closed subsets and d^+_{B} is the Hausdorff metric based on d_{B}
$(\mathcal{T}er(\Sigma,V), d_{\mathrm{t}})$	with d_{t} the term metric
$(\mathcal{T}er^\star(\Sigma,V), d^\star_{\mathrm{t}})$	$(\mathcal{T}er^\star(\Sigma,V), d^\star_{\mathrm{t}})$ is the metric completion of $(\mathcal{T}er(\Sigma,V), d_{\mathrm{t}})$ (see Chapter 2)

For arbitrary X with $s, t \in X^*$, the *Baire metric* d_{B} is defined as:

$$d_{\mathrm{B}}(s,t) = \begin{cases} 0 & \text{if } s = t \\ 2^{-k} & \text{if } s \neq t \text{ and } k = \max\{|u| \mid u \leqslant s \text{ and } u \leqslant t\} \end{cases}$$

A proof that the Baire metric is actually a metric can be found, e.g., in the textbook by De Bakker and De Vink [BV96].

It is easy to show the following:

$$\max\{|u| \mid u \leqslant s \text{ and } u \leqslant t\} = \max\{n \mid s[n] = t[n]\}$$

where $s[n]$ and $t[n]$ truncations of s and t (see Chapter 2). Moreover, if we consider elements of X^* to be terms over a signature in which each element of X occurs as a unary function symbol and in which a special nullary function symbol occurs to end strings, then we also have:

$$\max\{|u| \mid u \leqslant s \text{ and } u \leqslant t\} = \min\{|p| \mid s \text{ and } t \text{ conflict at } p \in \mathbb{N}^*\},$$

where s and t *conflict* at $p \in \mathbb{N}^*$ if $p \in \mathcal{P}os(s) \cap \mathcal{P}os(t)$ and $root(s|_p) \neq root(t|_p)$.

We have for each string s:

$$B(s, 2^{-(|s|+1)}) = \{s\}.$$

Hence, $X - \{s\}$ is closed and for each $Y \subseteq X^*$ we have that $Y = \mathrm{Cl}(Y)$. Thus, each $Y \subseteq X^*$ is closed.

Observe that $0 \leqslant d_{\mathrm{B}}(s,t) \leqslant 1$ for all strings s and t. Hence, we can define the Hausdorff metric based on d_{B}. As shown by De Bakker and De Vink [BV96, Lemma 2.9], the Hausdorff metric d_{H} based on d_{B} is equal to the metric d_{B}^+, which is defined as follows, given an alphabet X and sets $S, T \subseteq X^*$:

$$d_{\mathrm{B}}^+(S,T) = \begin{cases} 0 & \text{if } S = T \\ 2^{-k} & \text{if } S \neq T \text{ and } k = \max\{n \mid S[n] = T[n]\} \end{cases}$$

The metric space $(\wp_{\mathrm{nc}}(\mathbb{N}^*), d_{\mathrm{B}}^+)$ is not a complete metric space. The Cauchy sequence $(\{0^n\})_{n<\omega}$, e.g., would have the limit $\{0^\omega\}$, where the definition of 0^n is extended appropriately. However, $0^\omega \notin \mathbb{N}^*$.

We next define the term metric, a metric on terms. The metric is easily seen to correspond to the metric informally described in the introduction to the metric completion approach. Moreover, it extends the Baire metric to terms over arbitrary signatures.

Given $s, t \in \mathcal{T}er(\Sigma, V)$, the *term metric*, denoted d_{t}, is defined as:

$$d_{\mathrm{t}}(s,t) = \begin{cases} 0 & \text{if } s = t \\ 2^{-k} & \text{if } s \neq t \text{ and } k = \min\{|p| \mid s \text{ and } t \text{ conflict at } p \in \mathbb{N}^*\} \end{cases}$$

where s and t *conflict* at $p \in \mathbb{N}^*$ if $p \in \mathcal{P}os(s) \cap \mathcal{P}os(t)$ and $root(s|_p) \neq root(t|_p)$. A proof that d_{t} is a metric can be found in the work of Arnold and Nivat [AN80]. As in the case of the Baire metric, we have $0 \leqslant d_{\mathrm{t}}(s,t) \leqslant 1$ for all $s, t \in \mathcal{T}er(\Sigma, V)$.

3.4.2 Infinite Terms

We define the set of infinite terms by means of metric completion:

Definition 3.4.1. *The set* $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$ *is defined as* $\mathcal{T}er^{\star}(\Sigma, V)$.

The set $\mathcal{T}er^{\star}(\Sigma, V)$ is part of the metric space $(\mathcal{T}er^{\star}(\Sigma, V), d^{\star}_{\mathrm{t}})$, the metric completion of $(\mathcal{T}er(\Sigma, V), d_{\mathrm{t}})$. Thus, each element of $\mathcal{T}er^{\star}(\Sigma, V)$ is an equivalence class of Cauchy sequences over $\mathcal{T}er(\Sigma, V)$ (see Chapter 2).

By definition of metric completion, we have that $\mathcal{T}er(\Sigma, V)$ is isomorphic to a subset of $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$. Each member of the subset (an equivalence class) has a representative $(s_\kappa)_{\kappa<\omega}$ for which there exists an ordinal $\beta < \omega$ such that for all $\beta < \gamma_1, \gamma_2 < \omega$ we have $s_{\gamma_1} = s_{\gamma_2}$.

Example 3.4.2. In the case of Σ_{CL}, we can define I^{ω} as the following equivalence class of Cauchy sequences:

$$[\![(I^n)_{n<\omega}]\!] .$$

That $(I^n)_{n<\omega}$ is a Cauchy sequence follows immediately by the fact that:

$$\lim_{n\to\omega}(\min\{|p| \mid I^n \text{ and } I^{n+1} \text{ conflict at } p \in \mathbb{N}^*\}) = \omega .$$

As in the case of ideal completion and partial functions, we can show that I^{ω}, as defined above, is identical to the infinite term I^{ω} defined in Example 3.1.2. Of course, to show this, we first have to prove that $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$ is a final F_{Σ}-coalgebra. However, as in the case of partial functions, the proof is omitted, because it is very similar to the proof given in Example 3.2.20.

We next define the notions of root symbol, positions, subterms, and replacements of subterms for the elements of $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$. With respect to the representatives of the elements of $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$, we employ the notions of root symbol, positions, subterms, and replacements of subterms as defined for $\mathcal{T}er(\Sigma, V)$.

Definition 3.4.3. *Let* $S, T \in \mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$*, with* $(s_\kappa)_{\kappa<\omega}$ *a representative of* S *and* $(t_\kappa)_{\kappa<\omega}$ *a representative of* T.

- *The* root symbol *of* S*, denoted* $root(S)$*, is defined as:*

$$root(S) = \lim\ (root(s_\kappa))_{\kappa<\omega} ,$$

  with $(\Sigma \cup V, d_{\mathrm{d}})$ *the employed metric space.*
- *The set of* positions *of* S*, denoted* $\mathcal{P}os(S)$*, is defined as:*

$$\mathcal{P}os(S) = \lim\ (\mathcal{P}os(s_\kappa))_{\kappa<\omega} ,$$

  with $(\wp_{\mathrm{nc}}(\mathbb{N}^*), d_{\mathrm{H}})$ *the employed metric space.*
- *If* $p \in \mathcal{P}os(S)$*, then the* subterm *at position* p*, denoted* $S|_p$ *is defined as:*

$$S|_p = [\![(s_\kappa|^*_p)_{\kappa<\omega}]\!] ,$$

  with $(\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V), d^{\star}_{\mathrm{t}})$ *the employed metric space and with:*

$$s|^*_p = \begin{cases} t & \text{if } p \notin \mathcal{P}os(s) \text{ and } t \in \mathcal{T}er(\Sigma, V) \\ s|_p & \text{if } p \in \mathcal{P}os(s) \end{cases}$$

– *If $p \in \mathcal{P}os(S)$, then the* replacement *of the subterm at position p by T, denoted $S[T]_p$, is defined as:*

$$S[T]_p = [\![(s_\kappa[t_\kappa]^*_p)_{\kappa<\omega}]\!]\,,$$

with $(\mathcal{T}er^\infty_{\mathrm{m}}(\Sigma, V), d^\star_{\mathrm{t}})$ the employed metric space and with:

$$s[t]^*_p = \begin{cases} s & \text{if } p \notin \mathcal{P}os(s) \\ s[t]_p & \text{if } p \in \mathcal{P}os(s) \end{cases}$$

We have the following:

Proposition 3.4.4. *Let $S, T \in \mathcal{T}er^\infty_{\mathrm{m}}(\Sigma, V)$. The maps root and $\mathcal{P}os$ are well-defined, the set $\mathcal{P}os(S)$ is downward closed, and if $p \in \mathcal{P}os(S)$, then $S|_p, S[T]_p \in \mathcal{T}er^\infty_{\mathrm{m}}(\Sigma, V)$.*

Proof. Assume $(s_\kappa)_{\kappa<\omega}$ is a Cauchy sequence in the equivalence class of S. To prove the well-definedness of $root(S)$, observe there exist $\beta < \omega$ such that $d_{\mathrm{t}}(s_{\gamma_1}, s_{\gamma_2}) < \frac{1}{2}$ for all $\beta < \gamma_1, \gamma_2 < \omega$. This implies, by definition of d_{t}, that there exists a symbol $f \in \Sigma \cup V$ such that for all s_γ with $\beta < \gamma < \omega$ it holds that $root(s_\gamma) = f$. Hence, $(root(s_\kappa))_{\kappa<\omega}$ is a Cauchy sequence. Obviously, $\lim\,(root(s_\kappa))_{\kappa<\omega} = f$. Remark that the actual representative chosen is irrelevant by definition of $d^\star_{\mathrm{t}}$. Hence, $root(S)$ is well-defined.

To prove the well-definedness of $\mathcal{P}os(S)$, observe that the following value becomes arbitrary small for large enough $\gamma < \omega$, since $(s_\kappa)_{\kappa<\omega}$ is a Cauchy sequence:

$$d^+_{\mathrm{B}}\Big(\mathcal{P}os(s_\gamma), \bigcap_{\kappa>\gamma} \mathcal{P}os(s_\kappa)\Big)\,.$$

By the same fact, we have for all $\gamma_1 \leqslant \gamma_2$ that:

$$\bigcap_{\kappa_1>\gamma_1} \mathcal{P}os(s_{\kappa_1}) \subseteq \bigcap_{\kappa_2>\gamma_2} \mathcal{P}os(s_{\kappa_2})\,,$$

where the positions in $\bigcap_{\kappa_2>\gamma_2} \mathcal{P}os(s_{\kappa_2})$ and not in $\bigcap_{\kappa_1>\gamma_1} \mathcal{P}os(s_{\kappa_1})$ are at greater depths for larger values of γ_1 and γ_2. Hence, for large enough $\gamma < \omega$ we have that the following value becomes arbitrary small:

$$d^+_{\mathrm{B}}\Big(\bigcap_{\kappa>\gamma} \mathcal{P}os(s_\kappa), \bigcup_{\gamma'<\omega} \bigcap_{\kappa'>\gamma'} \mathcal{P}os(s_{\kappa'})\Big)\,.$$

By the triangle inequality we now have that the next value also becomes arbitrary small for large enough $\gamma < \omega$:

$$d^+_{\mathrm{B}}\Big(\mathcal{P}os(s_\gamma), \bigcup_{\gamma'<\omega} \bigcap_{\kappa'>\gamma'} \mathcal{P}os(s_{\kappa'})\Big)\,.$$

The above implies that:

$$\lim\,(\mathcal{P}os(s_\kappa))_{\kappa<\omega} = \bigcup_{\gamma'<\omega} \bigcap_{\kappa'>\gamma'} \mathcal{P}os(s_{\kappa'})\,.$$

The set $\lim (\mathcal{P}os(s_\kappa))_{\kappa<\omega}$ is a non-empty closed subset of $\mathbb{N}^*$: It is non-empty, because at least ϵ is in the set. It is closed, because each subset of $\mathbb{N}^*$ is closed in $(\wp_{\mathrm{nc}}(\mathbb{N}^*), d_{\mathrm{B}}^+)$. As in the case of $root(S)$, the actual representative chosen is irrelevant by definition of $d_{\mathrm{t}}^\star$. Hence, $\mathcal{P}os(S)$ is well-defined.

That $\mathcal{P}os(S)$ is downward closed follows immediately by the fact that s_κ is downward closed for each $\kappa < \omega$.

Suppose $(t_\kappa)_{\kappa<\omega}$ is a Cauchy sequence in the equivalence class of T. It follows immediately that $(s_\kappa|_p^*)_{\kappa<\omega}$ and $(s_\kappa[t_\kappa]_p^*)_{\kappa<\omega}$ are both Cauchy sequences. Hence, $[\![(s_\kappa|_p^*)_{\kappa<\omega}]\!]$ and $[\![(s_\kappa[t_\kappa]_p^*)_{\kappa<\omega}]\!]$ are defined. As before, the actual representatives chosen are irrelevant by definition of $d_{\mathrm{t}}^\star$. Hence, we have $S|_p, S[T]_p \in \mathcal{T}er_{\mathrm{m}}^\infty(\Sigma, V)$. □

Remark 3.4.5. In the case of $\mathcal{P}os(S)$ we can prove along similar lines as exhibited above that $\bigcap_{\gamma<\omega}\bigcup_{\kappa>\gamma}\mathcal{P}os(s_\kappa)$ is a limit of $(\mathcal{P}os(s_i))_{\kappa<\omega}$. As limits of sequences are unique in metric spaces, we have:

$$\bigcup_{\gamma<\omega}\bigcap_{\kappa>\gamma}\mathcal{P}os(s_\kappa) = \bigcap_{\gamma<\omega}\bigcup_{\kappa>\gamma}\mathcal{P}os(s_\kappa)\,.$$

3.4.3 Final Coalgebra

As in the case of ideal completion and partial functions, we define a map, denoted φ_{m}, to turn $\mathcal{T}er_{\mathrm{m}}^\infty(\Sigma_\perp, V)$ into an F_Σ-coalgebra:

$$\varphi_{\mathrm{m}}(S) = \begin{cases} x & \text{if } root(S) = x \in V \\ (f, S|_1, \ldots, S_n) & \text{if } root(S) = f \in \Sigma_n \end{cases}$$

where $S \in \mathcal{T}er_{\mathrm{m}}^\infty(\Sigma_\perp, V)$. As before, the map adheres to the informal description of φ_∞, as presented in Section 3.1.1.

The main theorem of this section is as follows:

Theorem 3.4.6. *The pair $(\mathcal{T}er_{\mathrm{m}}^\infty(\Sigma_\perp, V), \varphi_{\mathrm{m}})$ is a final F_Σ-coalgebra.*

We do not prove the above theorem, as Barr [Bar93, Theorem 3.2] already gives a proof.

By combining Theorem 3.4.6 with Theorem 3.3.7 and the fact that final coalgebras are unique up to isomorphisms, it follows that $\mathcal{T}er_{\mathrm{m}}^\infty(\Sigma, V)$ is *isomorphic* to $\mathcal{T}er_{\mathrm{f}}^\infty(\Sigma, V)$. A more direct proof occurs as Theorem 12.2.1 in the book by Terese [Ter03]. In addition, by adding a fresh nullary function symbol $\perp$ to the signature Σ, it follows by Theorem 3.2.19 that $\mathcal{T}er_{\mathrm{m}}^\infty(\Sigma_\perp, V)$ and $\mathcal{T}er_{\mathrm{f}}^\infty(\Sigma_\perp, V)$ are *isomorphic* to $\mathcal{T}er_{\mathrm{i}}^\infty(\Sigma_\perp, V)$.

3.4.4 Homomorphisms

We next prove that the definitions of positions, subterms, and replacements of subterms as given for $\mathcal{T}er_{\mathrm{m}}^\infty(\Sigma, V)$ are identical to the definitions given in Sections 3.1.2, 3.1.3, and 3.1.4 in case $\mathcal{T}er_{\mathrm{m}}^\infty(\Sigma, V)$ is employed as final F_Σ-coalgebra.

Positions. Define a map $\mathcal{P}os'$ from $\mathbf{1} + \mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$ to $\to \wp(\mathbb{N}^*)$:

$$\mathcal{P}os'(x) = \begin{cases} \emptyset & \text{if } x \in \mathbf{1} \\ \mathcal{P}os(x) & \text{if } x \in \mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V) \end{cases}$$

Obviously, $\mathcal{P}os(S) = \mathcal{P}os' \circ \mathrm{inr}(S)$ for $S \in \mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$. Hence, we only need to prove:

Lemma 3.4.7. *Let $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$ be employed as final coalgebra. The map:*

$$\mathcal{P}os' : \mathbf{1} + \mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V) \to \wp(\mathbb{N}^*)$$

is the unique homomorphism defined in Section 3.1.2.

Proof. Completely analogous to the proof of Lemma 3.2.21. □

Subterms. Define a map $\_|'_\_$ from $\mathcal{TP}$ to $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V) \times \mathcal{TP}$:

$$\_|'_\_(S, p) = (S|_p, (S, p))\,.$$

Obviously, $S|_p = \pi_1 \circ \_|'_\_(S, p)$ for $(S, p) \in \mathcal{TP}$ and we only need to prove:

Lemma 3.4.8. *Let $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$ be employed as final coalgebra. The map:*

$$\_|'_\_ : \mathcal{TP} \to \mathcal{T}er^{\infty}_{\mathrm{f}}(\Sigma, V) \times \mathcal{TP}$$

is the unique homomorphism defined in Section 3.1.3.

Proof. Completely analogous to the proof of Lemma 3.2.22. □

Replacements of Subterms. Define the map $\_[\_]'_\_$ from $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V) + \mathcal{TPT}$ to $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$:

$$\_[\_]'_\_(x) \begin{cases} x & \text{if } x \in \mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V) \\ S[T]_p & \text{if } x = (S, p, T) \in \mathcal{TPT} \end{cases}$$

Obviously, $S[T]_p = \_[\_]'_\_ \circ \mathrm{inr}(S, p, T)$ for all $(S, p, T) \in \mathcal{TPT}$. Hence, we only need to prove:

Lemma 3.4.9. *Let $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$ be employed as final coalgebra. The map:*

$$\_[\_]'_\_ : \mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V) + \mathcal{TPT} \to \mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$$

is the unique homomorphism defined in Section 3.1.4.

Proof. Completely analogous to the proof of Lemma 3.2.23. □

3.5 Infinite λ-Terms

In this section, we define infinite λ-terms, which we employ in Chapter 4. A subset of the infinite λ-terms is formed by the (finite) λ-terms:

Definition 3.5.1. *The set of* λ-terms*, denoted* Λ*, is inductively defined as:*

1. $x \in \Lambda$*, if* $x \in V$*,*
2. $\lambda x.s \in \Lambda$*, if* $x \in V$ *and* $s \in \Lambda$*, and*
3. $(s \cdot t) \in \Lambda$*, if* $s, t \in \Lambda$*.*

As usual, we write $(s\,t)$ instead of $(s \cdot t)$ and $\lambda x_1 \dots x_n.s$ instead of $\lambda x_1 \dots . \lambda x_n.s$. Moreover, assuming left-associativity of $\cdot$, we omit parentheses whenever possible. Finally, we say that a variable x is *free* in a term if it does not occur in a subterm of the form $\lambda x.s$.

Defining the set of infinite λ-terms, denoted Λ^∞, is trivial, given the theory developed in the previous sections. Ignoring α-equivalence, infinite λ-terms can be defined as infinite (first-order) terms over the following *infinite* signature:

$$\Sigma_\lambda = \{\lambda x \mid x \in V\} \cup \{\cdot\}\,,$$

with each λx a unary function symbol and where $\cdot$ a binary function symbol. That Σ_λ is infinite does not pose a problem: In the previous sections, no assumptions were made regarding the cardinality of the assumed signature.

Since Chapter 4 draws heavily on infinite λ-terms, we spell out the three representations infinite λ-terms, even though they follow trivially from the definitions in the previous sections. In Sections 3.5.1, 3.5.2, and 3.5.3, we respectively define infinite λ-terms by means of ideal completion, partial functions, and metric completion. In Section 3.5.4, we discuss α-equivalence.

Remark that the sets of infinite λ-terms defined by ideal completion, partial functions, and metric completion are isomorphic, given the material presented in the previous sections. That the signature is infinite, is again irrelevant.

Before proceeding with the definitions of infinite λ-terms, we spell out one more definition for (finite) λ-terms:

Definition 3.5.2. *The* set of positions *of a* λ*-term* s*, denoted* $\mathcal{P}os(s)$*, is inductively defined as:*

$$\mathcal{P}os(s) = \begin{cases} \{\epsilon\} & \text{if } s = x \in V \\ \{\epsilon\} \cup \{1 \cdot p \mid p \in \mathcal{P}os(s')\} & \text{if } s = \lambda x.s' \\ \{\epsilon\} \cup \bigcup_{i \in \{1,2\}} \{i \cdot p \mid p \in \mathcal{P}os(s_i)\} & \text{if } s = s_1 s_2 \end{cases}$$

Given a λ-term s, we use $s|_p$ to denote the subterm at position $p \in \mathcal{P}os(s)$. Moreover, by $s[t]_p$ we denote the replacement of the subterm at position $p \in \mathcal{P}os(s)$ by the λ-term t. The definitions are identical to those of first-order terms.

3.5.1 Ideal Completion

Adding a fresh nullary function symbol $\bot$ to Σ_λ, obtaining the signature $\Sigma_{\lambda,\bot} = \Sigma_\lambda \cup \{\bot\}$, we can de define the prefix order on $\Lambda_\bot = \mathcal{T}er(\Sigma_{\lambda,\bot}, V)$:

Definition 3.5.3. *The* prefix order *on* λ*-terms, denoted* $\preccurlyeq$*, is the smallest binary relation such that:*

1. $x \preccurlyeq x$ *for all* $x \in V$,
2. $\bot \preccurlyeq s$ *for all* $s \in \Lambda_\bot$,
3. $\lambda x.s \preccurlyeq \lambda x.t$, *if* $s \preccurlyeq t$ *and* $s, t \in \Lambda_\bot$, *and*
4. $s_1 s_2 \preccurlyeq t_1 t_2$, *if* $s_1 \preccurlyeq t_1$, $s_2 \preccurlyeq t_2$, *and* $s_1, s_2, t_1, t_2 \in \Lambda_\bot$.

The prefix order defines a CUSL on $\Lambda_\bot$, where each $\bot$ denote an *unspecified* λ-subterm.

We can now define infinite λ-terms by means of ideal completion:

Definition 3.5.4. *The set of* infinite λ-terms*, denoted* $\Lambda^\infty_{\mathrm{i},\bot}$*, is defined as:*

$$\Lambda^\infty_{\mathrm{i},\bot} = \{I \subseteq \Lambda_\bot \mid I \text{ is and ideal}\}\,.$$

The notions of root symbols, positions, subterms, and replacements of subterms are defined exactly as in Section 3.2.2. As explained in the same section, we have that $\Lambda_\bot$ is isomorphic to:

$$\{S \mid S \in \Lambda^\infty_{\mathrm{i},\bot} \text{ with } S \text{ finite}\} \subseteq \Lambda^\infty_{\mathrm{i},\bot}\,.$$

The isomorphism assigns to each λ-term its principal ideal.

Example 3.5.5. The two infinite λ-terms informally depicted in Figure 3.4 correspond respectively to the ideal:

$$\{\bot, \lambda x.\bot, \lambda xy.\bot, \lambda xyz.\bot, \ldots\}$$

and the ideal:

$$\downarrow\{(\lambda x.y)((\lambda x.y)(\lambda x.\bot)), \ldots\}\,.$$

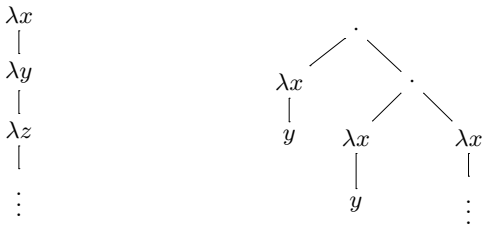

Figure 3.4. Two infinite λ-terms

Bibliographic Notes. Infinite λ-terms defined by means of ideal completion are used in numerous publications on Böhm-like trees for the λ-calculus. Among the publications are the ones by Berry [Ber78a], Hyland [Hyl75, Hyl76], Lévy [Lév75, Lév78], Wadsworth [Wad76, Wad78], and Welch [Wel75].

3.5.2 Partial Functions

We next define infinite λ-terms as partial functions:

Definition 3.5.6. *The set of infinite* λ*-terms, denoted* $\Lambda^\infty_{\mathrm{f}}$*, is the set of all partial functions* $S : \mathbb{N}^* \rightharpoonup (\Sigma \cup V)$ *satisfying:*

1. *the set of values on which S is defined is downward closed,*
2. *if $S(p) \in V$, then $S(q)$ is undefined for all $q > p$,*
3. *if $S(p) = \lambda x$, then $S(p \cdot i)$ is defined if $i = 1$ and undefined otherwise, and*
4. *if $S(p) = \cdot$, then $S(p \cdot i)$ is defined if $i \in \{1, 2\}$ and undefined otherwise,*

where $p \in \mathbb{N}^*$.

By restricting the set of infinite λ-terms to the terms that have a finite number of defined values, we obtain a set of terms that is isomorphic to the set Λ of (finite) λ-terms.

Example 3.5.7. The infinite λ-terms depicted in Figure 3.4 correspond respectively to:

$$\{\epsilon \mapsto \lambda x, 1 \mapsto \lambda y, 11 \mapsto \lambda z, \ldots\}$$

and to:

$$\{\epsilon \mapsto \cdot, 1 \mapsto \lambda x, 11 \mapsto y, 2 \mapsto \cdot, 21 \mapsto \lambda x, 22 \mapsto \lambda x, 211 \mapsto y, \ldots\}\,.$$

Bibliographic Notes. The standard reference on infinite λ-terms defined by means of partial functions is Barendregt's book [Bar84].

3.5.3 Metric Completion

In case of metric completion, recall that the term metric is defined as follows:

Definition 3.5.8. *Let $s, t \in \Lambda$. The* term metric*, denoted d_{t}, is defined as:*

$$d_{\mathrm{t}}(s,t) = \begin{cases} 0 & \text{if } s = t \\ 2^{-k} & \text{if } s \neq t \text{ and } k = \min\{|p| \mid s \text{ and } t \text{ conflict at } p \in \mathbb{N}^*\} \end{cases}$$

where s and t conflict *at $p \in \mathbb{N}^*$ if $p \in \mathcal{P}os(s) \cap \mathcal{P}os(t)$ and $root(s|_p) \neq root(t|_p)$.*

The infinite λ-terms are now defined as $\Lambda^\infty_{\mathrm{m}} = \Lambda^\star$, where $(\Lambda^\star, d^\star_{\mathrm{t}})$ is the metric completion of $(\Lambda, d_{\mathrm{t}})$.

A number of subsets of $\Lambda^\infty_{\mathrm{m}}$ are defined by Kennaway, Klop, Sleep, and De Vries [KKSV97]. As they show, these subsets can be defined by parameterising the definition of the term metric. The parameterisation is as follows, where we slightly generalise the definition by Kennaway, Klop, Sleep, and De Vries:

Definition 3.5.9. *Let $s, t \in \Lambda$. The* term metric*, denoted d^l_{t}, is defined as:*

$$d^l_{\mathrm{t}}(s,t) = \begin{cases} 0 & \text{if } s = t \\ 2^{-k} & \text{if } s \neq t \text{ and } k = \min\{l(p,s) \mid s \text{ and } t \text{ conflict at } p \in \mathbb{N}^*\} \end{cases}$$

where s and t conflict *at $p \in \mathbb{N}^*$ if $p \in \mathcal{P}os(s) \cap \mathcal{P}os(t)$ and $root(s|_p) \neq root(t|_p)$ and where $l : \mathbb{N}^* \times \Lambda \to \mathbb{N}$ is such that for all $p \in \mathcal{P}os(s)$:*

- *if s and t do not conflict at any $q \leqslant p$, then $l(p,s) = l(p,t)$, and*
- *if $q \leqslant p$, then $l(s,q) \leqslant l(s,p)$.*

That d^l_{t} is a metric is easy to prove.

With the help of parameterisation and a number of criteria for the 'good' behaviour of infinite λ-terms, four instantiations of d_{t}^{l} are singled out by Kennaway, Klop, Sleep, and De Vries. Employing notation similar to that of Severi and De Vries [SV02a] and assuming $s \in \Lambda$ and $p \in \mathcal{P}os(s)$, the four instantiations are as follows:

Term Metric	**Map**
d_{t}^{f}	$f(p,s) = l^{000}(p,s)$
d_{t}^{u}	$u(p,s) = l^{001}(p,s)$
d_{t}^{w}	$w(p,s) = l^{101}(p,s)$
d_{t}^{t}	$t(p,s) = l^{111}(p,s)$

where it is assumed that:

$$l^{abc}(\epsilon, s) = 0$$

$$l^{abc}(i \cdot p, s) = \begin{cases} a + l^{abc}(p, s') & \text{if } s = \lambda x.s' \text{ and } i = 1 \\ b + l^{abc}(p, s_1) & \text{if } s = s_1 s_2 \text{ and } i = 1 \\ c + l^{abc}(p, s_2) & \text{if } s = s_1 s_2 \text{ and } i = 2 \end{cases}$$

Remark that $f(p,s) = 0$, independent of p and s and that $t(p,s) = |p|$.

As explained by Kennaway, Klop, Sleep, and De Vries, we have that each of the metric completions of the spaces defined by the above metrics yields a subset of $\Lambda_{\mathrm{m}}^{\infty}$:

Term Metric	**Subset**
d_{t}^{f}	$\Lambda_{f}^{\infty} = \Lambda$
d_{t}^{u}	Λ_{u}^{∞}
d_{t}^{w}	Λ_{w}^{∞}
d_{t}^{t}	$\Lambda_{t}^{\infty} = \Lambda_{\mathrm{m}}^{\infty}$

Above, the set Λ_{u}^{∞} is the set of infinite λ-terms in which no subterms of the forms $\lambda x_1.\lambda x_2.\ldots.\lambda x_n.\ldots$ and $((\ldots(\ldots S_n)\ldots)S_2)S_1$ occur. Moreover, the set Λ_{w}^{∞} is the set of infinite λ-terms in which no subterms of the form $((\ldots(\ldots S_n)\ldots)S_2)S_1$ occur.

It is easy to see that the following relations hold between the sets of infinite terms:

$$\Lambda = \Lambda_{f}^{\infty} \subseteq \Lambda_{u}^{\infty} \subseteq \Lambda_{w}^{\infty} \subseteq \Lambda_{t}^{\infty} = \Lambda_{\mathrm{m}}^{\infty}\,.$$

Bibliographic Notes. The basic reference on the definition of infinite λ-terms by means of metric completion is the paper by Kennaway, Klop, Sleep, and De Vries [KKSV97]. Some of the details as presented in that paper can also be found in Chapter 12 of the book by Terese [Ter03].

3.5.4 α-Equivalence

We next define α-equivalence for infinite λ-terms. The definition depends on a notion of α-equivalence for (finite) λ-terms, which we define first (see also Barendregt's book [Bar84, Definition 2.1.11 and Convention 2.1.12]).

To define α-equivalence for *finite* λ-terms, we first define a map from $\Lambda \times V \times V$ to Λ, denoted $s[x \mapsto z]$:

$$s[x \mapsto z] = \begin{cases} z & \text{if } s = x \text{ with } x \in V \\ y & \text{if } s = y \text{ with } y \in V \text{ and } y \neq x \\ \lambda x.s' & \text{if } s = \lambda x.s' \\ \lambda y.(s'[x \mapsto z]) & \text{if } s = \lambda y.s' \text{ and } y \neq x \\ (s_1[x \mapsto z])(s_2[x \mapsto z]) & \text{if } s = s_1 s_2 \end{cases}$$

where $s \in \Lambda$ and $z \notin \mathcal{V}ar_a(s)$, with:

$$\mathcal{V}ar_a(s) = \begin{cases} \{x\} & \text{if } s = x \text{ with } x \in V \\ \{x\} \cup \mathcal{V}ar_a(s') & \text{if } s = \lambda x.s' \\ \mathcal{V}ar_a(s_1) \cup \mathcal{V}ar_a(s_2) & \text{if } s = s_1 s_2 \end{cases}$$

Two terms $s, t \in \Lambda$ are now said to be *α-equivalent*, denoted $s =_\alpha t$ if one following holds:

- $s = t$ with $s, t \in V$,
- $s = \lambda x.s'$, $t = \lambda y.t'$, and $s'[x \mapsto z] =_\alpha t'[y \mapsto z]$ for $z \notin \mathcal{V}ar_a(s') \cup \mathcal{V}ar_a(t')$, or
- $s = s_1 s_2$, $t = t_1 t_2$, and $s_1 =_\alpha t_1$ and $s_2 =_\alpha t_2$.

As usual in the λ-calculus, we implicitly consider λ-terms *modulo* α-equivalence.

To define α-equivalence with respect to infinite λ-terms, we add to Σ_λ a fresh nullary function symbol $\mho$. With the help of $\mho$, we can define a prefix order on the infinite terms over $\Sigma_{\lambda,\mho}$ analogous to what we did in the case of ideal completion. Denoting by $\Lambda^\infty_\mho$ the infinite λ-terms over $\Sigma_{\lambda,\mho}$ we next define:

Definition 3.5.10. *Let $S, T \in \Lambda^\infty_\mho$. The infinite λ-terms S and T are said to be* α-equivalent*, denoted $S =_\alpha T$, if and only if each finite prefix of S is α-equivalent to some finite prefix of T.*

Although a finite prefix is not a (finite) λ-term (over $\Sigma_{\lambda,\mho}$), we have that the set of finite prefixes is isomorphic to the set of λ-terms. Hence, under assumption of the isomorphism that assigns to each λ-term its principal ideal, the above definition suffices.

Remark that the set of infinite λ-terms Λ^∞ is isomorphic to a subset of $\Lambda^\infty_\mho$. Hence, by the above definition, we also know whether two infinite λ-terms are α-equivalent or not. Like in the case of (finite) λ-terms, we *implicitly* consider infinite λ-terms modulo α-equivalence in the following chapters.

Remark 3.5.11. Two observations are to be made with respect to the above definition of α-equivalence:

- Alternatively to considering infinite λ-terms modulo α-equivalence it is also possible to define infinite λ-terms based on De Bruijn-indices [Bru72]. This is already noted by Kennaway, Klop, Sleep, and De Vries [KKSV97].
- The above approach of dealing with α-equivalence falls outside the categorical approach as described in Section 3.1. To capture α-equivalence in the categorical

approach, the work by Fiore, Plotkin, and Turi [FPT99] is relevant. They provide a categorical definition of finite λ-terms that takes into account α-equivalence. Unfortunately, it is not possible to simply dualise their approach: Dualisation of the functors they define only allows for infinite λ-terms with a *finite* number of *free* variables.
Other relevant work is that by Gabbay and Pitts [GP02]. Their approach easily combines with our categorical approach to allow variable binding and α-equivalence. However, the underlying category employed by Gabbay and Pitts is not the category of sets and total functions. It is the Fraenkel-Mostowski permutation model of set theory. In this model, the Axiom of Choice is no longer valid.

4

λ-Calculus

"Listen you semi-evolved simian," cut in Zaphod,
"go climb a tree will you?"

— Douglas Adams
The Restaurant at the End of the Universe (1980)

In this chapter, we survey the Böhm-like trees that have been defined for the $\lambda\beta$-*calculus*. We also present a number of the trees that have been defined for two interesting systems extending the $\lambda\beta$-calculus: PCF and the $\lambda\beta\eta$-calculus.

It not the aim of this chapter to present every Böhm-like tree that has ever been defined for any system extending the $\lambda\beta$-calculus. As such, no further mention is made, e.g., of the Böhm-like tree defined by Dezani-Ciancaglini, Severi, and De Vries [DCSV03]. Moreover, since this chapter has the character of a survey, all proofs are omitted. They can be found in the cited literature.

As explained in Chapter 1, a Böhm-like tree of a term s is a sort of normal form of s which takes into account infinite reductions. More precisely, it takes into account *maximal* reductions, where a reduction is maximal if it is either a reduction to normal form or an infinite reduction.

Two problems were observed regarding maximal reductions: Terms may become infinitely large and, no matter how often a term is reduced, its root may always become a redex again, i.e., it may never become *root-stable*. As we explain next, both these problems can be observed in the $\lambda\beta$-calculus.

To understand that terms may become infinitely large, suppose that $\mathbf{Y}$ is a λ-term that behaves like a fixed-point combinator, i.e., $\mathbf{Y}s \to_\beta^* s(\mathbf{Y}s)$. We now have the following reduction:

$$\mathbf{Y}(\lambda xy.x) \to_\beta^* \lambda y_1.\mathbf{Y}(\lambda xy.x) \to_\beta^* \lambda y_1 y_2.\mathbf{Y}(\lambda xy.x) \to_\beta^* \cdots .$$

Obviously, the further this reduction progresses the larger the terms become. Hence, considering reductions to be a limit process, the final λ-term will be *infinite*.

To understand that λ-terms may never become root stable, consider the λ-term $\Omega = (\lambda x.xx)(\lambda x.xx)$. Starting with Ω, only the following reduction is possible:

$$(\lambda x.xx)(\lambda x.xx) \to_\beta (\lambda x.xx)(\lambda x.xx) \to_\beta (\lambda x.xx)(\lambda x.xx) \to_\beta \cdots .$$

Hence, Ω reduces only to itself and not reduce to a root-stable term.

As remarked in Chapter 1, the above two problems may respectively be solved by introducing infinite terms and a fresh nullary function symbol $\bot$. Infinite λ-terms, which are the infinite terms required in the case of the $\lambda\beta$-calculus, are introduced in the last section of Chapter 3. The usage of the fresh function symbol $\bot$ is detailed in the following sections.

Besides the above two problems, there is an additional problem not discussed in Chapter 1. To understand this problem, consider the following reduction:

$$\Omega(\mathbf{Y}(\lambda xy.x)) \to_\beta \Omega(\mathbf{Y}(\lambda xy.x)) \to_\beta \Omega(\mathbf{Y}(\lambda xy.x)) \to_\beta \cdots ,$$

where the Ω-redex is contracted in each step. Although repeatedly contracting Ω yields a maximal reduction, it is not a *fair* reduction: The redex $\mathbf{Y}(\lambda xy.x)$ is ignored forever. Hence, continuously contracting Ω says not all there is to say about the shape of $\Omega(\mathbf{Y}(\lambda xy.x))$ after other maximal reductions. To overcome this problem we consider maximal *fair* reductions instead of just maximal reductions. Here, fair means that no redex that occurs along a reduction is ignored forever. That is, each redex is either contracted or erased at some point in the reduction.

Having highlighted all problems we will run into while defining the Böhm-like trees of the $\lambda\beta$-calculus and related systems, we next give some preliminary definitions related to the $\lambda\beta$-calculus in Section 4.1. Thereafter, in Section 4.2, we define the Böhm-like trees of the $\lambda\beta$-calculus. The properties these trees have in common are discussed in Section 4.3. In Section 4.4, we present two alternative, but equivalent, ways of defining the Böhm-like trees of the $\lambda\beta$-calculus. Finally, in Sections 4.5 and 4.6 we discuss a number of Böhm-like trees for PCF and the $\lambda\beta\eta$-calculus, respectively.

4.1 Preliminaries

The $\lambda\beta$-calculus defines only the following rewrite rule:

$$(\lambda x.s)t \to_\beta s[x := t] ,$$

where $(\lambda x.s)t$ is called a *β-redex* and where $s[x := t]$ is inductively defined as:

$$s[x := t] = \begin{cases} t & \text{if } s = x \in V \\ y & \text{if } s = y \in V \text{ and } y \neq x \\ \lambda z.((s'[y \mapsto z])[x := t]) & \text{if } s = \lambda y.s' \text{ and } z \notin \mathcal{V}ar_a(s') \cup \mathcal{V}ar_a(t) \\ (s_1[x := t])(s_2[x := t]) & \text{if } s = s_1 s_2 \end{cases}$$

Above, $\lambda z.((s'[y \mapsto z])[x := t])$ is often written as $\lambda y.(s'[x := t])$, where it is implicitly assumed that capturing of free variables is avoided (see also Convention 2.1.12 in Barendregt's book [Bar84]). The definitions of $s[x \mapsto z]$ and $\mathcal{V}ar_a(s)$ can be found in Section 3.5.4.

In the remainder of this chapter, we distinguish between three sets of λ-terms. The sets encompass respectively the *head normal forms*, the *weak head normal forms* and the *root-stable terms* and are defined as follows:

Definition 4.1.1. *Let s be a (finite) λ-term.*

1. *If $s = \lambda x_1 \dots x_m.y s_1 \dots s_n$, then s is a* head normal form*,*
2. *If $s = \lambda x.s'$ or $s = x s_1 \dots s_n$, then s is a* weak head normal from*.*
3. *If s does not reduce to a β-redex, then s is a* root-stable term*.*

By the above definition, it follows immediately that any head normal form is a weak head normal form and that any weak head normal form is root-stable. These

implications cannot be reversed. For example, $\lambda x.\Omega$ is a weak head normal form but not a head normal form, as $root(\lambda x.\Omega|_{11}) = root(\lambda x.xx) \notin V$. Moreover, $\Omega\Omega$ is root-stable but not a weak head normal form, as $root(\Omega\Omega) \neq \lambda x$ and $root(\Omega\Omega|_{11}) = root(\lambda x.xx) \notin V$.

4.2 Böhm-Like Trees

We define *three* Böhm-like trees for the $\lambda\beta$-calculus: the Böhm trees, the Lévy-Longo or lazy trees, and the Berarducci trees. Recall from Chapter 1 that the generic name 'Böhm-like tree' derives from the most well-known of the three trees: the Böhm tree.

Assuming that infinite λ-terms are defined by means of ideal completion, each of the three Böhm-like trees is defined according to a three-step pattern:

1. A map ω from $\Lambda_\perp$ to $\Lambda_\perp$, called a *direct approximant function*, is defined.
2. For each $s \in \Lambda_\perp$ a set, called the *auxiliary set* of s, is defined:
$$\mathcal{A}(s) = \{\omega(t) \mid s \to_\beta^* t\}\,,$$
where ω is the direct approximant function defined in the first step.
3. The *Böhm-like tree* of $s \in \Lambda$ is defined:
$$\mathrm{BLT}(s) = {\downarrow}\mathcal{A}(s)\,,$$
where ${\downarrow}\mathcal{A}(s)$ denotes the downward closure of $\mathcal{A}(s)$.

Regarding the first step, we call the value $\omega(s)$ obtained by applying a direct approximant function ω to s the *direct approximant* of s. In the case of the three direct approximant functions we present below, it holds that the direct approximant of s is a *root-stable prefix* of s. That is, the direct approximant is a term $t \preccurlyeq s$ such that for all $t|_p \neq \perp$ we have that $s|_p$ is a root-stable subterm of s.

Regarding the second step, we have that the auxiliary set of s contains for each reduction starting in s the direct approximant of the final term of the reduction. Hence, all terms that occur along *maximal fair reductions* of s are considered while defining the auxiliary set of s. This implies that auxiliary sets help to overcome the fairness problem identified in the introduction of this chapter.

Remark that auxiliary sets do not need to be ideals. For example, we have for all three Böhm-like trees of the $\lambda\beta$-calculus that $\mathcal{A}(\lambda x.y) = \{\lambda x.y\}$, as follows easily from the discussion below. Obviously, $\{\lambda x.y\}$ is not an ideal, since $\perp \notin \{\lambda x.y\}$.

Regarding the third step, we have that ${\downarrow}\mathcal{A}(s)$ is an ideal whenever $\mathcal{A}(s)$ is directed. Hence, in this particular case we have that $\mathrm{BLT}(s)$ is an infinite λ-term. As we will see below, it holds for each of the three Böhm-like trees of the $\lambda\beta$-calculus that the auxiliary set is directed.

The Böhm-like trees of the $\lambda\beta$-calculus only represent subterms that become root-stable, as required with respect to the root-stability problem mentioned in the introduction of this chapter. To see that only root-stable subterms are represented, recall from above that each direct approximant function only represents a root-stable prefix and observe that root-stable prefixes can only become larger along

reductions. If a root-stable prefix would become smaller, then a root-stable term would become a β-redex again, which is impossible by definition of root-stability.

Remark 4.2.1. Given a direct approximant function ω, some authors, e.g., Hyland [Hyl75] and Wadsworth [Wad76, Wad78], define the Böhm-like tree for each λ-term s as:

$$\mathrm{BLT}(s) = \{t' \mid s \to_\beta^* t,\, t' \preccurlyeq \omega(t)\}\,.$$

That is, the creation of auxiliary sets and taking there downward closure is combined into a single step.

Given a λ-term s, the direct approximant functions of the three Böhm-like trees of the $\lambda\beta$-calculus are defined as follows:

Böhm Direct Approximant The Böhm direct approximant of s, denoted by $\omega_{\mathrm{B}}(s)$, is the greatest prefix of s, such that $\omega_{\mathrm{B}}(s) \preccurlyeq s[\bot]_p$ for all $p \in \mathcal{P}os(s)$ with $s|_p$ not a head normal form.

Lévy-Longo Direct Approximant The Lévy-Longo direct approximant of s, denoted by $\omega_{\mathrm{LL}}(s)$, is the greatest prefix of s, such that $\omega_{\mathrm{LL}}(s) \preccurlyeq s[\bot]_p$ for all $p \in \mathcal{P}os(s)$ with $s|_p$ not a weak head normal form.

Berarducci Direct Approximant The Berarducci direct approximant of s, denoted by $\omega_{\mathrm{Be}}(t)$, is the greatest prefix of s, such that $\omega_{\mathrm{Be}}(s) \preccurlyeq s[\bot]_p$ for all $p \in \mathcal{P}os(s)$ with $s|_p$ not root-stable.

Since subterms in head normal form and weak head normal form are root-stable, as remarked in Section 4.1, it follows that each direct approximant is a root-stable prefix.

The Böhm direct approximant of a term s can also be defined as the unique normal form of s with respect to the following *confluent* and *terminating* rewrite system:

$$\begin{aligned}(\lambda x.s)t &\to_\omega \bot\\ \bot t &\to_\omega \bot\\ \lambda x.\bot &\to_\omega \bot\end{aligned}$$

In case of the Lévy-Longo direct approximant, a confluent and terminating rewrite system can also be defined: It consists of the first two of the above three rewrite rules. With regard to the Berarducci direct approximant no finite rewrite system exists that is confluent and terminating: Confluence and termination of such a rewrite system would imply that root-stability is decidable, which it is not.

Example 4.2.2. Consider the following λ-term:

$$s = \lambda x.x(\lambda y.\Omega)(\bot\bot)\,,$$

where $\Omega = (\lambda x.xx)(\lambda x.xx)$. The values of the three direct approximants of s are:

$$\begin{aligned}\omega_{\mathrm{B}}(t) &= \lambda x.x\bot\bot\\ \omega_{\mathrm{LL}}(t) &= \lambda x.x(\lambda y.\bot)\bot\\ \omega_{\mathrm{Be}}(t) &= \lambda x.x(\lambda y.\bot)(\bot\bot)\end{aligned}$$

In the above example, we have:

$$\omega_{\mathrm{B}}(s) \preccurlyeq \omega_{\mathrm{LL}}(s) \preccurlyeq \omega_{\mathrm{Be}}(s)\,.$$

This relation holds for all λ-terms: Recall that every head normal form is a weak head normal form, but not the other way around, and also recall that every weak head normal form is root-stable, and again not the other way around.

Given the three-step approach explained above, auxiliary sets and Böhm-like trees can be defined based on each of the three direct approximant functions. For each λ-term s, the auxiliary sets and Böhm-like trees are as follows:

$$\begin{aligned} \mathcal{A}_{\mathrm{B}}(s) &= \{\omega_{\mathrm{B}}(t) \mid s \to_\beta^* t\} & \mathrm{BLT}_{\mathrm{B}}(s) &= {\downarrow}\mathcal{A}_{\mathrm{B}}(s) \\ \mathcal{A}_{\mathrm{LL}}(s) &= \{\omega_{\mathrm{LL}}(t) \mid s \to_\beta^* t\} & \mathrm{BLT}_{\mathrm{LL}}(s) &= {\downarrow}\mathcal{A}_{\mathrm{LL}}(s) \\ \mathcal{A}_{\mathrm{Be}}(s) &= \{\omega_{\mathrm{Be}}(t) \mid s \to_\beta^* t\} & \mathrm{BLT}_{\mathrm{Be}}(s) &= {\downarrow}\mathcal{A}_{\mathrm{Be}}(s) \end{aligned}$$

Since root-stable prefixes can only become larger and since the $\lambda\beta$-calculus is confluent, it follows that the above auxiliary sets are directed. Hence, each Böhm-like tree is an element of $\Lambda_{\mathrm{i},\bot}^{\infty}$, i.e., an infinite λ-term defined by ideal completion.

Given a λ-term s, we call the infinite λ-term $\mathrm{BLT}_{\mathrm{B}}(s)$ the *Böhm tree* of s, the infinite λ-term $\mathrm{BLT}_{\mathrm{LL}}(s)$ the *Lévy-Longo tree* or *lazy tree* of s, and $\mathrm{BLT}_{\mathrm{Be}}(s)$ the *Berarducci tree* of s.

Example 4.2.3. Consider the following λ-term:

$$s = (\lambda z_1.\lambda z_2.\lambda x.x z_1 z_2)(\lambda y.\Omega)(\bot\bot)$$

which reduces to the λ-term:

$$\lambda x.x(\lambda y.\Omega)(\bot\bot)\,,$$

as employed in Example 4.2.2. The three Böhm-like trees of s are as follows:

$$\begin{aligned} \mathrm{BLT}_{\mathrm{B}}(s) &= {\downarrow}\{\bot, \lambda x.x\bot\bot\} \\ \mathrm{BLT}_{\mathrm{LL}}(s) &= {\downarrow}\{\bot, \lambda x.x(\lambda y.\bot)\bot\} \\ \mathrm{BLT}_{\mathrm{Be}}(s) &= {\downarrow}\{\bot, \lambda x.x(\lambda y.\bot)(\bot\bot)\} \end{aligned}$$

Similar to what holds in case of the direct approximants, we have in the above example that:

$$\mathrm{BLT}_{\mathrm{B}}(s) \preccurlyeq \mathrm{BLT}_{\mathrm{LL}}(s) \preccurlyeq \mathrm{BLT}_{\mathrm{Be}}(s)\,.$$

This relation holds for all λ-terms, which is an immediate consequence of the fact that for every λ-term s it holds that $\omega_{\mathrm{B}}(s) \preccurlyeq \omega_{\mathrm{LL}}(s) \preccurlyeq \omega_{\mathrm{Be}}(s)$ and the definitions of auxiliary sets and downward closure.

The next example shows that the Böhm-like trees can become infinite:

Example 4.2.4. Consider the following λ-term:

$$s = \Theta(\lambda f.\lambda x.xf)\,,$$

where Θ is Turing's fixed-point combinator, i.e., $\Theta = AA$ with $A = \lambda xy.y(xxy)$ (see, e.g., Definition 6.1.4 in Barendregt's book [Bar84]). With respect to s we have the following reduction:

$$s \to_\beta^* \lambda x_1.x_1 s \to_\beta^* \lambda x_1.x_1(\lambda x_2.x_2 s) \to_\beta^* \cdots$$

Since each term $\lambda x_i.x_i t$, with t some arbitrary λ-term, is a head normal form, we have for s and all three of defined Böhm-like trees that the Böhm-like tree of s is equal to:

$$\downarrow\{\bot, \lambda x_1.x_1\bot, \lambda x_1.x_1(\lambda x_2.x_2\bot), \ldots\}\,.$$

As there is no bound on the size of the reducts of s, the Böhm-like tree of s must be infinite.

Bibliographic Notes. As remarked by Barendregt [Bar84], Böhm trees were suggested by the original proof of Böhm's theorem [Böh68]. The definition of Böhm trees as presented above, occurs in the work of Barendregt [Bar84], Hyland [Hyl75], Lévy [Lév78], Wadsworth [Wad76, Wad78], and Welch [Wel75]. Barendregt also provides an alternative coalgebraic definition, which is explained in Section 4.4. The rewrite system that defines the Böhm direct approximants is discussed both by Barendregt [Bar84] and Lévy [Lév78].

Lévy-Longo trees were first defined by Lévy [Lév75], who employs the above definition. Longo [Lon83] gives an alternative coalgebraic definition, as explained in Section 4.4. Longo [Lon83] also proves numerous properties regarding Lévy-Longo trees. The rewrite system that defines the Lévy-Longo direct approximants is discussed by Lévy [Lév75].

Finally, Berarducci trees were first defined by Berarducci [Ber96], employing infinitary rewriting, as explained in Section 4.4.

4.3 Common Properties

We divide the properties the three Böhm-like trees of the $\lambda\beta$-calculus have in common into two categories: The properties of the direct approximants and the properties of the Böhm-like trees. The next two sections each deal with one of the two categories.

4.3.1 Direct Approximants

The three direct approximant functions, as defined for the $\lambda\beta$-calculus, have in common the following three properties, which are related to root-stability:

1. $\omega(s) \preccurlyeq s$,
2. if $s|_p$ is a β-redex with $p \in \mathcal{P}os(s)$, then $\omega(s) \preccurlyeq s[\bot]_p$, and
3. if $s \to_\beta t$, the $\omega(s) \preccurlyeq \omega(t)$,

where $s, t \in \Lambda_\bot$ and where ω is one of the three direct approximant functions.

The first property states that a direct approximant of a λ-term is a prefix of that term. This property holds, as direct approximants are root-stable prefixes. The second property states that a β-redex cannot be a subterm of a direct approximant. This property holds, as β-redexes are not root-stable. Note that the first property is implied by the second one in case a β-redex occurs in s. The third property states that direct approximants cannot become smaller along reductions. As before, this property holds by root-stability.

It is readily proved from the three properties, that any direct approximant that adheres to the properties must represent a root-stable prefix. In the case of a λ-term in normal form, this follows by the first property. For every other λ-term, this follows by the second and third property.

The Böhm and Lévy-Longo direct approximants also have in common the following property:

4. if $s \preccurlyeq t$, then $\omega(s) \preccurlyeq \omega(t)$.

This property is a consequence of the fact that head normal forms and weak head normal forms are preserved by the prefix order. We say that Böhm and the Lévy-Longo direct approximants are *monotone* with respect to the prefix order.

Remark 4.3.1. The Berarducci direct approximant is not monotone with respect to the prefix order. To see this, consider the λ-terms $\bot\bot$ and $\Omega = (\lambda x.xx)(\lambda x.xx)$ and remark:

$$\bot\bot \preccurlyeq \Omega\,,$$

The Berarducci direct approximants of these terms are respectively $\bot\bot$ and $\bot$. Hence, we have:

$$\omega_{\mathrm{Be}}(\bot\bot) = \bot\bot \not\preccurlyeq \bot = \omega_{\mathrm{Be}}(\Omega)\,.$$

Given the rewrite systems that define the Böhm and Lévy-Longo direct approximants, we can also identify a number of common properties. Denoting by $d \to_\omega e$ an arbitrary rewrite rule of one of the rewrite systems, we obtain the following:

1. $e = \bot$,
2. $\bot$ is a normal form,
3. $s \to_\omega^= \bot$ for all $s \preccurlyeq d$, and
4. $(\lambda x.s)t \to_\omega \bot$.

Confluence and termination follow immediately from the properties, as do the four properties of the Böhm and Lévy-Longo direct approximant functions when we define the direct approximant of a term as its unique normal form.

4.3.2 Böhm-Like Trees

We next give an overview of properties the three Böhm-like trees of the $\lambda\beta$-calculus have in common. The properties are summarised in Table 4.1. A checkmark (✓) indicates that a certain property holds, while a dash (-) indicates that a certain property does not hold.

Table 4.1. Properties of the three Böhm-like trees

Property	Böhm-Like Tree		
	$\mathrm{BLT_B}$	$\mathrm{BLT_{LL}}$	$\mathrm{BLT_{Be}}$
Preservation	✓	✓	✓
Congruence	✓	✓	✓
Monotonicity	✓	✓	–
Continuity	✓	✓	–
Syntactic Continuity	✓	✓	–
Sequentiality	✓	✓	–

We next discuss each of the properties in the table. Moreover, we indicate whether each of the properties is implied by the properties of the direct approximant functions as discussed in the previous section. Of course, if a property is not implied, ground exists to try to established additional properties. No attempt to establish additional properties is made here; this is postponed to subsequent chapters.

Preservation. A property shared between all three Böhm-like trees is *preservation under rewriting*. Given λ-terms s and t, the property states:

$$s \to_\beta^* t \text{ implies } \mathrm{BLT}(s) = \mathrm{BLT}(t)\,.$$

The validity of the property follows directly from the confluence of the $\lambda\beta$-calculus and the third property of direct approximants functions mentioned in the previous section.

Congruence. A second property of all three Böhm-like trees is *congruence* of Böhm-like tree equality. That is, given λ-terms s and t and a context $C[\Box]$, it holds that:

$$\mathrm{BLT}(s) = \mathrm{BLT}(t) \text{ implies } \mathrm{BLT}(C[s]) = \mathrm{BLT}(C[t])\,.$$

The property does not follow from the properties mentioned in the previous section. However, it does follow immediately whenever Böhm-like trees are defined by means of infinitary rewriting, as discussed in Section 4.4.

Monotonicity and Continuity. Given two λ-terms s and t, *monotonicity* states that:

$$s \preccurlyeq t \text{ implies } \mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)\,.$$

Moreover, *continuity* states that:

$$\mathrm{BLT}(s) = \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq s\}\,.$$

Remark 4.3.2. The above formulation of continuity derives from usual formulation that occurs in theory of partial orders, which states that BLT should be monotonic and that:

$$\mathrm{BLT}(\bigsqcup D) = \bigsqcup\{\mathrm{BLT}(d) \mid d \in D\}\,,$$

where D is any directed set that has a least upper bound. That our formulation derives from the usual one is immediate by monotonicity and the observation that $\bigsqcup\{t \mid t \preccurlyeq s\} = s$.

For both Böhm and Lévy-Longo trees, monotonicity and continuity follows by monotonicity of the direct approximants with respect to the prefix order and by left-linearity of the $\lambda\beta$-calculus.

Remark 4.3.3. Monotonicity and continuity do not hold for Berarducci trees, as noted, e.g., by Dezani-Ciancaglini, Severi, and De Vries [DCSV03]. To see that monotonicity does not hold, consider the λ-terms $\bot\bot$ and $\Omega = (\lambda x.xx)(\lambda x.xx)$, which are also considered in Remark 4.3.1. Although it holds that:

$$\bot\bot \preccurlyeq \Omega\,,$$

the Berarducci trees of the λ-terms are respectively $\{\bot, \bot\bot\}$ and $\{\bot\}$. Hence, we have:

$$\mathrm{BLT}_{\mathrm{Be}}(\bot\bot) = \{\bot, \bot\bot\} \not\preccurlyeq \{\bot\} = \mathrm{BLT}_{\mathrm{Be}}(\Omega)\,.$$

To see that continuity does not hold for Berarducci trees, consider again the λ-term Ω. It holds that:

$$\{s \mid s \preccurlyeq \Omega\} = {\downarrow}\{\bot(\lambda x.xx), (\lambda x.xx)\bot, \Omega\}\,,$$

Moreover, the following holds:

$$\begin{aligned}\mathrm{BLT}_{\mathrm{Be}}(\Omega) &= {\downarrow}\{\bot\}\\ \mathrm{BLT}_{\mathrm{Be}}(\bot(\lambda x.xx)) &= {\downarrow}\{\bot(\lambda x.xx)\}\\ \mathrm{BLT}_{\mathrm{Be}}((\lambda x.xx)\bot) &= {\downarrow}\{\bot\bot\}\end{aligned}$$

Hence, we have:

$$\mathrm{BLT}_{\mathrm{Be}}(\Omega) = {\downarrow}\{\bot\} \neq {\downarrow}\{\bot, \bot\bot, \bot(\lambda x.xx)\} = \bigsqcup\{\mathrm{BLT}_{\mathrm{Be}}(s) \mid s \preccurlyeq \Omega\}\,.$$

Syntactic Continuity. Given a λ-term s and a context $C[\Box]$, *syntactic continuity* states:

$$\mathrm{BLT}(C[s]) = \bigsqcup\{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}\,.$$

Remark the subtle difference with continuity: $t \in \mathrm{BLT}(s)$ is employed instead of $t \preccurlyeq s$. Irrespective of this subtle difference, the formulation of syntactic continuity also derives from the concept of continuity as it is occurs in theory of partial orders. An explanation of this can be found, e.g., in the book by Amadio and Curien [AC98]. Essentially, it is to be understood as the continuity of contexts.

Syntactic continuity of Böhm and Lévy-Longo trees does not follow from the properties established for direct approximants. A property which does imply syntactic continuity is discussed in Chapter 6.

Remark 4.3.4. Syntactic continuity does not hold for Berarducci trees. To see this, consider the λ-term $\lambda x.xx$ and the context $C[\Box] = \Box(\lambda x.xx)$. It holds that:

$$\mathrm{BLT}_{\mathrm{Be}}(\lambda x.xx) = {\downarrow}\{\lambda x.xx\}$$

and that:

$$\mathrm{BLT}_{\mathrm{Be}}(C[\lambda x.xx]) = \mathrm{BLT}_{\mathrm{Be}}(\Omega) = \downarrow\{\bot\}\,.$$

Hence, we have:

$$\bigsqcup\{\mathrm{BLT}_{\mathrm{Be}}(C[s]) \mid s \in \mathrm{BLT}_{\mathrm{Be}}(\lambda x.xx)\} = \downarrow\{\bot(\lambda x.xx)\}$$

and we also have:

$$\downarrow\{\bot\} \neq \downarrow\{\bot(\lambda x.xx)\}\,.$$

Sequentiality. Given a λ-term s and a position $p \in \mathcal{P}os(\mathrm{BLT}(s))$, where $\mathrm{BLT}(s)$ is either the Böhm or Lévy-Longo tree of s, *sequentiality* states that $\mathrm{BLT}(s)|_p = \{\bot\}$ implies exactly one of the following:

1. for all λ-terms t, if $t \succcurlyeq s$, then $\mathrm{BLT}(t)|_p = \{\bot\}$, or
2. there exists a unique position $q \in \mathcal{P}os(s)$ with $s|_q = \bot$ such that for all λ-terms t, if $t \succcurlyeq s$ and $\mathrm{BLT}(t)|_p \neq \{\bot\}$, then $t|_q \neq \bot$.

A further explanation of sequentiality can be found in the introduction of Chapter 8. Similar to syntactic continuity, sequentiality does not follow from the properties established thus far with respect to direct approximants.

Sequentiality was first shown to hold for Böhm trees by Berry [Ber78a]. A proof of sequentiality of Böhm trees also occurs in Barendregt's book [Bar84]. For Lévy-Longo trees, a proof of sequentiality is easily constructed out of any of the proofs for Böhm trees.

Remark 4.3.5. To see that sequentiality does not hold for Berarducci trees, it is important to realise that sequentiality assumes for all $t \succcurlyeq s$ that $p \in \mathcal{P}os(\mathrm{BLT}(t))$. This property does not hold in the case of Berarducci trees, as it depends on monotonicity.

4.4 Alternative Definitions

In this section, we discuss two alternative definitions of each of the three Böhm-like trees of the λβ-calculus: the coalgebraic definition and the infinitary rewriting definition.

Both the coalgebraic definition and the infinitary rewriting definition yield Böhm-like trees that are identical to the Böhm-like trees of the approach presented in Section 4.2. That the trees are identical is readily proved using the fact that head normal forms, weak head normal forms, and root-stability re-occur in all three definitions.

4.4.1 Coalgebra

Coalgebraically, Böhm-like trees are defined according to the following pattern, where s is a λ-term and where P is a property only satisfiable by root-stable λ-terms:

1. If s is not reducible to a term satisfying P, then the root of the Böhm-like tree of s is equal to $\bot$.
2. If s is reducible to a term t satisfying P, then the Böhm-like tree of s is T. Here, T is constructed out of t by replacing those subterms of t that do not satisfy P by their Böhm-like trees, where the Böhm-like trees of the subterms are constructed in the same way as the Böhm-like tree of s.

Remark that a Böhm-like tree is defined in terms of itself. In other words, the definition is coalgebraic. Remark too that all subterms to be replaced by their Böhm-like trees are considered in parallel. Hence, as all subterms that satisfy P are root-stable, it holds that all maximal fair reductions are considered.

Given that the coalgebraic definition requires infinite λ-terms to be partial functions, the three Böhm-like trees of the $\lambda\beta$-calculus are as defined follows:

Böhm Tree. Let $s \in \Lambda_\bot$. If s is not reducible to a head normal form, then define:

$$\mathrm{BLT_B}(s)(\epsilon) = \bot\,.$$

Otherwise, if s is reducible to the head normal form $t = \lambda x_1 \ldots x_m.yt_1 \ldots t_n$, then define:

$$\mathrm{BLT_B}(s)(p) = \begin{cases} t(p) & \text{if } p \not> q \text{ with } t|_q = t_i \text{ for some } i \\ \mathrm{BLT_B}(t_i)(r) & \text{if } p = q \cdot r \text{ with } t|_q = t_i \text{ for some } i \end{cases}$$

Lévy-Longo Tree. Let $s \in \Lambda_\bot$. If s is not reducible to a weak head normal form, then define:

$$\mathrm{BLT_{LL}}(s)(\epsilon) = \bot\,.$$

Otherwise, if s is reducible to the weak head normal form $t = \lambda x.t'$, then define:

$$\mathrm{BLT_{LL}}(s)(p) = \begin{cases} t(p) & \text{if } p \not> q \text{ with } t|_q = t' \\ \mathrm{BLT_{LL}}(t')(r) & \text{if } p = q \cdot r \text{ with } t|_q = t' \end{cases}$$

Finally, s is reducible to the weak head normal form $t = xt_1 \ldots t_n$, then define:

$$\mathrm{BLT_{LL}}(s)(p) = \begin{cases} t(p) & \text{if } p \not> q \text{ with } t|_q = t_i \text{ for some } i \\ \mathrm{BLT_{LL}}(t_i)(r) & \text{if } p = q \cdot r \text{ with } t|_q = t_i \text{ for some } i \end{cases}$$

Berarducci Tree. Let $s \in \Lambda_\bot$. If s is not reducible to a root-stable λ-term, then define:

$$\mathrm{BLT_{Be}}(s)(\epsilon) = \bot\,.$$

Otherwise, if s is reducible to a root-stable λ-term t, then define:

$$\mathrm{BLT_{Be}}(s)(p) = \begin{cases} t(p) & \text{if } t|_p \text{ root-stable} \\ \mathrm{BLT_{Be}}(t|_q)(r) & \text{if } p = q \cdot r \text{ with } t|_q \text{ not root-stable} \\ & \quad \text{and } t|_{q'} \text{ root-stable for all } q' < q. \end{cases}$$

Given $s \in \Lambda_\bot$, it is readily shown by induction on the length of the positions that all three Böhm-like trees define an element of $\Lambda^\infty_{\mathrm{f},\bot}$.

Example 4.4.1. Consider once more the λ-term from Example 4.2.3:

$$s = (\lambda z_1.\lambda z_2.\lambda x.x z_1 z_2)(\lambda y.\Omega)(\bot\bot)\,.$$

In the current approach, the three Böhm-like trees of the λ-term are as follows:

$$\begin{aligned}
\mathrm{BLT_B}(s) &= S \cup \{112 \mapsto \bot, 12 \mapsto \bot\} \\
\mathrm{BLT_{LL}}(s) &= S \cup \{112 \mapsto \lambda y., 1121 \mapsto \bot, 12 \mapsto \bot\} \\
\mathrm{BLT_{Be}}(s) &= S \cup \{112 \mapsto \lambda y., 1121 \mapsto \bot, 12 \mapsto \cdot, 121 \mapsto \bot, 122 \mapsto \bot\}
\end{aligned}$$

where:

$$S = \{\epsilon \mapsto \lambda x., 1 \mapsto \cdot, 11 \mapsto \cdot, 111 \mapsto x\}\,.$$

Bibliographic Notes. The coalgebraic definitions of Böhm trees and Lévy-Longo trees occur respectively in Barendregt's book [Bar84] and Longo's work [Lon83]. That the definitions are actually coalgebraic definitions is remarked by Jacobs and Rutten [JR97].

4.4.2 Infinitary Rewriting

To explain the infinitary rewriting definition of Böhm-like trees, we first need to define what a reduction is in infinitary rewriting. This is covered by the following definition, where it is assumed that the infinite λ-terms are defined by means of metric completion:

Definition 4.4.2. *A* transfinite reduction *of ordinal length α is a sequence of infinite λ-terms $(S_\kappa)_{\kappa<\alpha+1}$ such that $S_\kappa \to S_{\kappa+1}$ for all $\kappa < \alpha$. For each rewrite step $S_\kappa \to S_{\kappa+1}$, let d_κ denote the depth of the contracted redex. The reduction is called* weakly convergent *or* Cauchy convergent *if it is continuous in the sense of Definition 2.1.2. Furthermore, it is called* strongly convergent *if it is weakly convergent and if d_κ tends to infinity as κ approaches γ from below.*

With respect to the above definition, substitution for infinite λ-terms must be defined appropriately. This is done by Kennaway, Klop, Sleep, and De Vries [KKSV97].

Note that every weakly convergent reduction is strongly convergent. Moreover, note that all finite reductions, as defined in Chapter 2, are weakly convergent; the same cannot be said of all infinite reductions, as not every infinite reduction needs to have a limit, which is required here.

The three Böhm-like trees are defined according to the following four-step pattern:

1. A set $\mathcal{U} \subseteq \Lambda^\infty_{\mathrm{m},\bot}$ is defined, which is assumed to contain all infinite λ-terms whose Böhm-like tree is $\bot$.
2. A set of rewrite rules is defined:

   $$\{(\lambda x.S)T \to_\beta S[x := T]\} \cup \{S \to_\bot \bot \mid S \in \mathcal{U}\}\,.$$

   That is, the β-rule is lifted to infinite λ-terms and a rule is added for each $S \in \mathcal{U}$ that reduces S to $\bot$.
3. It is shown that the rewrite relation based on the strongly convergent reductions of rewrite system defined in the previous step is confluent and normalising.

4. Böhm-like trees of infinite λ-terms are defined to be the unique normal forms with respect to the strongly convergent reductions, where the unique normal forms exist by confluence and normalisation.

The above four-step pattern is slightly more general than both the other definitions of Böhm-like trees: the trees are not just defined for the (finite) λ-terms, but for all elements of $\Lambda^{\infty}_{\mathrm{m},\bot}$. Congruence of Böhm-like tree equality, as described in Section 4.3, is now immediate, since confluence implies:

$$C[s] \twoheadrightarrow \mathrm{BLT}(C[s]) = \mathrm{BLT}(C[\mathrm{BLT}(s)]) \twoheadleftarrow C[\mathrm{BLT}(s)] \twoheadleftarrow C[s]\,,$$

where $\twoheadrightarrow$ denotes a strongly convergent reduction and where BLT denotes the defined Böhm-like tree.

As shown by Kennaway, Van Oostrom, and De Vries [KOV99], it holds that the normalising strongly convergent reductions in the above four-step pattern are maximal fair, as long as subterms that never become root-stable are not counted towards fairness.

The sets $\mathcal{U}$ for the three Böhm-like trees of the $\lambda\beta$-calculus are defined as follows:

Böhm Tree $\mathcal{U}_{\mathrm{B}} = \{s \mid s \text{ does not reduce to a head normal form}\}$
Lévy-Longo Tree $\mathcal{U}_{\mathrm{LL}} = \{s \mid s \text{ does not reduce to a weak head normal form}\}$
Berarducci Tree $\mathcal{U}_{\mathrm{Be}} = \{s \mid s \text{ does not reduce to a root-stable term}\}$

Kennaway, Van Oostrom, and De Vries [KOV99] show that each of the above sets gives rise to an (infinitary) confluent and normalising rewrite system.

Example 4.4.3. Consider again the λ-term from Example 4.2.3:

$$s = (\lambda z_1.\lambda z_2.\lambda x.x z_1 z_2)(\lambda y.\Omega)(\bot\bot)$$

Ignoring the fact that infinite λ-terms defined by metric completion are actually equivalence classes of Cauchy sequences, we have that the three Böhm-like trees of the λ-term are as follows:

$$\begin{aligned}\mathrm{BLT}_{\mathrm{B}}(s) &= \lambda x.x\bot\bot\\ \mathrm{BLT}_{\mathrm{LL}}(s) &= \lambda x.x(\lambda y.\bot)\bot\\ \mathrm{BLT}_{\mathrm{Be}}(s) &= \lambda x.x(\lambda y.\bot)(\bot\bot)\end{aligned}$$

where the reductions to normal form are:

$$\begin{aligned}s &\to^*_\beta \lambda x.x(\lambda y.\Omega)(\bot\bot) \to^*_\bot \lambda x.x\bot\bot\\ s &\to^*_\beta \lambda x.x(\lambda y.\Omega)(\bot\bot) \to^*_\bot \lambda x.x(\lambda y.\bot)\bot\\ s &\to^*_\beta \lambda x.x(\lambda y.\Omega)(\bot\bot) \to^*_\bot \lambda x.x(\lambda y.\bot)(\bot\bot)\end{aligned}$$

Bibliographic Notes. The definition of Böhm and Lévy-Longo trees based on infinitary rewriting is by Kennaway, Klop, Sleep, and De Vries [KKSV97]. In their work, Böhm and Lévy-Longo trees are defined respectively for the infinite λ-terms

in the sets $\Lambda^{\infty}_{u,\bot}$ and $\Lambda^{\infty}_{w,\bot}$ (see Section 3.5.3). The extension to $\Lambda^{\infty}_{\mathrm{m},\bot}$ is by Kennaway, Van Oostrom, and De Vries [KOV99]. The definition of Berarducci trees based on infinitary rewriting is by Berarducci [Ber96].

The general four-step pattern sketched above is by Kennaway, Van Oostrom, and De Vries [KOV99] (see also Chapter 7). They show that it is possible to formulate a number of properties regarding the set $\mathcal{U}$, such that confluence and normalisation are implied whenever the properties are satisfied.

4.5 PCF

In this section and the next, we discuss a number of Böhm-like trees defined for PCF and the $\lambda\beta\eta$-calculus. Both PCF and the $\lambda\beta\eta$-calculus are extensions of the $\lambda\beta$-calculus.

To be exact, PCF is actually an extension of the *simply typed* $\lambda\beta$-calculus. It has special types representing natural numbers and Booleans. Moreover, besides the β-rule, PCF has a fixed-point rule and a number of rules that only apply to natural numbers and Booleans.

Ignoring the types, terms are defined as follows for PCF:

Definition 4.5.1 (Terms). *Given a countably infinite set of variables V and the set of natural numbers $\mathbb{N}$, the set of* PCF terms *is inductively defined as:*

1. *x is a PCF term, if $x \in V$,*
2. *$\lambda x.s$ is a PCF term, if $x \in V$ and s a PCF term,*
3. *$(s \cdot t)$ is a PCF term, if s and t are PCF terms,*
4. *n is a PCF term, if $n \in \mathbb{N}$, and*
5. **Y**, **succ**, **pred**, **zero**, **cond**, **true**, *and* **false** *are PCF terms.*

Thus, PCF terms are terms over the following signature:

$$\Sigma_{\mathrm{PCF}} = \{\lambda x \mid x \in V\} \cup \{\cdot\} \cup \{n \mid n \in \mathbb{N}\} \\ \cup \{\mathbf{Y}, \mathbf{succ}, \mathbf{pred}, \mathbf{zero}, \mathbf{cond}, \mathbf{true}, \mathbf{false}\}\,,$$

where each λx is unary, $\cdot$ is binary, and all other function symbols are nullary. Just as in the case of λ-terms, we usually omit $\cdot$ whenever it occurs in a PCF term and we assume left-associativity of $\cdot$. Moreover, we assume to work modulo α-equivalence (see Section 3.5.4).

The *infinite PCF terms* are defined by means of ideal completion, which requires a fresh nullary function symbol $\bot$ to be added to the signature. We omit the definition, as it easily derived from the material in Chapter 3.

The rewrite rules of PCF are as follows, where types are again omitted:

$$(\lambda x.s)t \to s[x := t]$$
$$\mathbf{Y}s \to s\,(\mathbf{Y}s)$$
$$\mathbf{succ}\; n \to n+1$$
$$\mathbf{pred}\;(n+1) \to n$$

$$\begin{aligned}
\mathbf{pred}\ 0 &\to 0\\
\mathbf{zero}\ (n+1) &\to \mathbf{false}\\
\mathbf{zero}\ 0 &\to \mathbf{true}\\
\mathbf{cond}\ \mathbf{true}\ s\ t &\to s\\
\mathbf{cond}\ \mathbf{false}\ s\ t &\to t
\end{aligned}$$

Above, $n \in \mathbb{N}$ and $+$ denotes addition of natural numbers.

We next discuss two Böhm-like trees for PCF: one that covers the whole of PCF and one that covers only a fragment.

PCF Böhm Tree. The *PCF Böhm tree* that occurs in the work by Amadio and Curien [AC98] is defined along the same lines as the three Böhm-like trees of the $\lambda\beta$-calculus in Section 4.2. That is, a direct approximant function is defined followed by the definition of auxiliary sets and Böhm trees.

As in the case of the Böhm and Lévy-Longo trees, the direct approximant function, denoted $\omega_{\mathbf{PCF}}$ can be defined by means of confluent and terminating rewrite system. The rules of this system can be divided into two categories: those with $\bot$ as their right-hand side and those with another right-hand side.

The rules with $\bot$ as their right-hand side are as follows:

$$\begin{aligned}
(\lambda x.s)t &\to_\omega \bot & \mathbf{pred}\ \bot &\to_\omega \bot\\
\bot s &\to_\omega \bot & \mathbf{zero}\ n &\to_\omega \bot\\
\mathbf{Y}s &\to_\omega \bot & \mathbf{zero}\ \bot &\to_\omega \bot\\
\mathbf{succ}\ n &\to_\omega \bot & \mathbf{cond}\ \mathbf{true}\ s\ t &\to_\omega \bot\\
\mathbf{succ}\ \bot &\to_\omega \bot & \mathbf{cond}\ \mathbf{false}\ s\ t &\to_\omega \bot\\
\mathbf{pred}\ n &\to_\omega \bot & \mathbf{cond}\ \bot\ s\ t &\to_\omega \bot
\end{aligned}$$

where $n \in \mathbb{N}$. Remark that the above rewrite rules include for each left-hand side of a PCF rewrite rule $l \to r$ a rule $l \to_\omega \bot$.

There is only one rule whose right-hand side is not $\bot$:

$$\mathbf{cond}\ (\mathbf{cond}\ s\ t_1\ t_2)\ t_1'\ t_2' \to_\omega \mathbf{cond}\ s\ (\mathbf{cond}\ t_1\ t_1'\ t_2')\ (\mathbf{cond}\ t_2\ t_1'\ t_2')\,.$$

It is easy to show that the rewrite system that consists of the above rewrite rules is confluent and terminating and that $\bot$ is a normal form. Given these facts, the *direct approximant* of a PCF term s, denoted $\omega_{\mathrm{PCF}}(s)$, is defined as the unique normal form with respect to the rewrite system.

The direct approximant of a PCF term s does not need to be a prefix of s, because of the rewrite rule whose right-hand side is not $\bot$. This is different from what we have in the case of the direct approximant functions of the $\lambda\beta$-calculus. Consequently, it does *not* hold that the PCF Böhm tree (partially) represents the root-stable part as created along maximal fair reductions. Adding the last rewrite rule to PCF instead of to the direct approximant rewrite system overcomes this problem.

The definitions of the auxiliary set and the PCF Böhm tree are now simply copies of the definitions for the three Böhm-like trees of the $\lambda\beta$-calculus. That is, the *auxiliary set* is defined for each PCF terms s as:

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

and the *PCF Böhm tree* is defined as:

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

Remark 4.5.2. The Böhm tree defined by Amadio and Curien [AC98] actually differs slightly from the Böhm tree defined here. In their work, a number of Böhm trees are identified by means of what they call an *extensional collapse*. The extensional collapse allows them to prove *full abstraction*.

Böhm-Like Tree for Fragment of PCF. Ong [Ong95] defines a Böhm tree for the fragment of PCF which only has the following rewrite rules:

$$(\lambda x.s)t \to s[x := t]$$
$$\mathbf{Y}s \to s\ (\mathbf{Y}s)$$

The direct approximant function is again defined by means of a rewrite system:

$$(\lambda x.s)t \to_\omega \bot$$
$$\lambda x.\bot \to_\omega \bot$$
$$\bot s \to_\omega \bot$$
$$\mathbf{Y}s \to_\omega \bot$$

Hence, with respect to the fragment, the rewrite rules are those defined above with the addition of a rule for the term $\lambda x.\bot$. As before, the rewrite system is confluent and terminating and direct approximants are defined as the unique normal forms.

The auxiliary set and Böhm-like tree are defined as usual.

Remark 4.5.3. Due to the rewrite rule $\lambda x.\bot \to_\omega \bot$, Ong's Böhm-like tree is comparable to the Böhm tree of the $\lambda\beta$-calculus, while the absence of this rewrite rule makes the PCF Böhm tree of Amadio and Curien more comparable to the Lévy-Longo tree.

Bibliographic Notes. PCF, or the Programming language for Computable Functions, was first formulated as a rewrite system by Plotkin [Plo77]. The language is based on the simply typed λ-calculus as formulated by Church [Chu40]. An overview of PCF and its properties can be found in Ong's handbook chapter [Ong95].

4.6 $\lambda\beta\eta$-Calculus

The $\lambda\beta\eta$-calculus extends the $\lambda\beta$-calculus with the η-rule:

$$\lambda x.sx \to_\eta s \qquad (x \text{ not free in } s)$$

With respect to this extended version of the $\lambda\beta$-calculus at least two Böhm-like trees exist. Both trees are constructed by defining an equivalence relation on one of the Böhm-like trees of the $\lambda\beta$-calculus and by working modulo the equivalence.

The defined equivalence relations are called the finite η-expansion and the infinite η-expansion, where η-expansion is the reverse of η-reduction:

$$s \to_{\bar{\eta}} \lambda x.sx \qquad (x \text{ not free in } s)$$

The following two sections each deal with one of the equivalence relations.

4.6.1 Finite η-Expansion

Finite η-expansion, denoted $=_{\bar{\eta}}$, is based on the Böhm tree of the $\lambda\beta$-calculus. Given $s, t \in \Lambda_\perp$, the expansion is defined as the smallest equivalence relation such that $\mathrm{BLT_B}(s) =_{\bar{\eta}} \mathrm{BLT_B}(t)$ if:

1. both $\mathrm{BLT_B}(s) = \{\perp\}$ and $\mathrm{BLT_B}(t) = \{\perp\}$, or
2. if $s \to_\beta^* \lambda x_1 \ldots x_m.ys_1 \ldots s_k$ and $t \to_\beta^* \lambda x_1 \ldots x_n.zt_1 \ldots t_l$ with $k \leqslant l$, then:
   - $(m - k) = (n - l)$,
   - $y = z$, and
   - $\lambda x_1 \ldots x_m.ys_1 \ldots s_k \to_{\bar{\eta}}^* \lambda x_1 \ldots x_n.ys_1 \ldots s_k s'_{k+1} \ldots s'_l$ such that:
     – $\mathrm{BLT_B}(s_i) =_{\bar{\eta}} \mathrm{BLT_B}(t_i)$ for all $1 \leqslant i \leqslant k$, and
     – $t_j \to_\beta^* s'_j$ for all $k + 1 \leqslant j \leqslant l$.

In the above definition α-equivalence is implicitly taken into account. The name *finite η-expansion* derives from the third item of the last clause: a finite number of η-expansions is applied to the head normal form of s.

Given $s \in \Lambda_\perp$, the Böhm-like tree $\mathrm{BLT}_{\bar{\eta}}(s)$ is defined as the equivalence class of $\mathrm{BLT_B}(s)$ with respect to the finite η-expansion.

Alternative Definition. With the help of infinitary rewriting, Severi and De Vries [SV02a] give an alternative definition of the Böhm-like tree defined by finite η-expansion. The main advantage of the infinitary rewriting approach is that each Böhm-like tree is represented by a unique infinite term instead of an equivalence class. Moreover, no equivalence relation is needed.

Contrary to the approach taken above, the infinitary rewriting approach employs η-reduction and not η-expansion. The rewrite rules are as follows:

$$\begin{aligned} (\lambda x.S)T &\to_\beta S[x := T] \\ \lambda x.Sx &\to_\eta S && (x \text{ not free in } S) \\ S &\to_\perp \perp && (S \in \mathcal{U}_\mathrm{B}) \end{aligned}$$

where $S, T \in \Lambda^\infty_{u,\perp}$ and where $\mathcal{U}_\mathrm{B}$ is the set of terms without head normal form.

As shown by Severi and De Vries confluence and normalisation hold with respect to strongly convergent reduction sequences based on the above rewrite rules. They also show that confluence would be lost if $S, T \in \Lambda^\infty_\perp$ was assumed instead of $S, T \in \Lambda^\infty_{u,\perp}$. The Böhm-like trees are defined as the unique normal forms of the λ-terms.

4.6.2 Infinite η-Expansion

Like finite η-expansion, *infinite η-expansion*, denoted $=_{\eta!}$, is based on the Böhm trees of the $\lambda\beta$-calculus. Given $s, t \in \Lambda_\perp$, it is defined as the smallest equivalence relation such that $\mathrm{BLT_B}(s) =_{\eta!} \mathrm{BLT_B}(t)$ if:

1. both $\mathrm{BLT_B}(s) = \{\perp\}$ and $\mathrm{BLT_B}(t) = \{\perp\}$, or
2. if $s \to_\beta^* \lambda x_1 \ldots x_m.y s_1 \ldots s_k$ and $t \to_\beta^* \lambda x_1 \ldots x_n.z t_1 \ldots t_l$ with $k \leqslant l$, then:
   - $(m - k) = (n - l)$,
   - $y = z$, and
   - $\lambda x_1 \ldots x_m.y s_1 \ldots s_k \to_{\bar{\eta}}^* \lambda x_1 \ldots x_n.y s_1 \ldots s_k x_{m+1} \ldots x_n$ such that:
     - $\mathrm{BLT_B}(s_i) =_{\eta!} \mathrm{BLT_B}(t_i)$ for all $1 \leqslant i \leqslant k$, and
     - $\mathrm{BLT_B}(x_j) =_{\eta!} \mathrm{BLT_B}(t_j)$ for all $m + 1 \leqslant j \leqslant n$.

As before, the definition implicitly takes into account α-equivalence. The name *infinite η-expansion* derives from the third item of the second clause: a finite number of η-expansions is applied to each head normal form of s, and this process is repeated for the variables that are created during the η-expansion. Due to the repetition there can occur an infinite number of η-expansions for each head normal form and the result can be called an infinite η-expansion.

Similar to finite η-expansion, given $s \in \Lambda_\perp$, the Böhm-like tree $\mathrm{BLT}_{\eta!}(s)$ is defined as equivalence class of $\mathrm{BLT_B}(s)$ with respect to the infinite η-expansion.

Alternative Definition. As in the case of finite η-expansion, there is a paper by Severi and De Vries [SV02b] which provides an alternative definition based on infinitary rewriting. In this case, the rewrite rules are as follows:

$$\begin{aligned} (\lambda x.S)T &\to_\beta S[x := T] \\ \lambda x.S\,T &\to_{\eta!} S && (x \twoheadrightarrow_{\bar{\eta}} T \text{ and } x \text{ not free in } S) \\ S &\to_\perp \perp && (S \in \mathcal{U}_\mathrm{B}) \end{aligned}$$

where $S, T \in \Lambda^\infty_{u,\perp}$. Here, $\twoheadrightarrow_{\bar{\eta}}$ denotes a strongly convergent reduction consisting of only η-expansions.

Severi and De Vries show that confluence and normalisation hold with respect to strongly convergent reduction sequences based on the above rewrite rules. Moreover, they also show that confluence would be lost if $S, T \in \Lambda^\infty_\perp$ was assumed instead of $S, T \in \Lambda^\infty_{u,\perp}$. Again, the Böhm-like trees are defined as the unique normal forms of the λ-terms.

Bibliographic Notes. The Böhm-like tree based on infinite η-expansion was first defined by Hyland [Hyl75, Hyl76]. It also occurs implicitly in the work of Wadsworth [Wad76]. An alternative definition, based on infinitely branching trees, and not presented here, occurs in the work of Nakajima [Nak75].

5

Böhm-Like Trees[†]

He could sense, too, the thrill of being a tree,
which was something he hadn't expected.

— Douglas Adams
So Long, and Thanks For All the Fish (1984)

In this chapter, we define Böhm-like trees for TRSs. The definition follows the three-step pattern for the Böhm-like trees of the $\lambda\beta$-calculus as described in Section 4.2. The definition of the direct approximant function is based on the three properties established in Section 4.3.1. As such, all Böhm-like trees for TRSs partially represent the root-stable part of a term as created along maximal fair reductions, similar to the Böhm-like trees of the $\lambda\beta$-calculus.

With respect to the TRSs considered in this chapter, and also in the next chapters, one very important restriction is imposed:

All considered TRSs are left-linear.

Non-left-linear rules allow us to observe if two or more terms are equal, even in case those terms are *root-active*. As such, root-active terms can contribute to the root-stable part of a term as created along maximal fair reductions. Since it is rather peculiar to have root-stability depend on root-activeness in this way, i.e., root-stability depends on terms which by themselves can never become root-stable, we choose to consider only left-linear systems. For similar reasons, left-linear systems are the only systems considered by Kennaway, Van Oostrom, and De Vries [KOV99].

This chapter is structured as follows: We start in Section 5.1 by connecting partial terms with TRSs. In Section 5.2, we define Böhm-like trees for TRSs. In Section 5.3, we discuss monotonicity and continuity of Böhm-like trees. In Section 5.4, a class of direct approximant functions is defined which always yields Böhm-like trees that are monotone and continuous. The class of Böhm-like trees is defined by means of a class of TRSs. Finally, in Section 5.5, we discuss related work.

In the remainder of this chapter we assume that Σ is an arbitrary signature and that V is a countably infinite set of variables. Moreover, to explain certain aspects of the concepts that are introduced, the following rewrite rules from Combinatory Logic (CL) are employed in some instances (see also Barendregt's book [Bar84] or the book by Terese [Ter03]):

$$\begin{aligned} Sxyz &\to xz(yz) \\ Kxy &\to x \\ Ix &\to x \end{aligned}$$

[†]This chapter is partially based on earlier work by the author [Ket04].

The above three rewrite rules form an orthogonal TRS. Hence, left-linearity and confluence are implied.

5.1 Partial Terms

Given a TRS $\mathcal{R} = (\Sigma, R)$ we can define a TRS $\mathcal{S} = (\Sigma_\perp, R)$, where $\Sigma_\perp = \Sigma \cup \{\perp\}$. The definition of $\mathcal{S}$ is sound with respect to the rewrite rules of $\mathcal{R}$, as $\Sigma \subseteq \Sigma_\perp$. Moreover, $\mathcal{S}$ has the same confluence and termination properties as $\mathcal{R}$, which follows immediately by considering $\perp$ to be a variable which we have singled out.

We define root-stable prefixes for TRSs:

Definition 5.1.1. *Let $\mathcal{R} = (\Sigma, R)$ be a TRS and $s, t \in \mathcal{T}er(\Sigma_\perp, V)$. The term t is a* root-stable prefix *of s, given that $t \preccurlyeq s$ and such that for all $t|_p \neq \perp$ with $p \in \mathcal{P}os(t)$ it holds that $s|_p$ is a root-stable subterm of s.*

The following property holds with respect to the extension of the prefix order to substitutions. The property plays an essential rôle in Section 5.4, where it helps to establish, under certain assumptions, that $s \preccurlyeq t$ with $t \to t'$ implies $s \to^= s'$ with $s' \preccurlyeq t'$

Lemma 5.1.2. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$ with t linear. If $s \preccurlyeq \tau(t)$ for some substitution τ, then there exist a term $s' \in \mathcal{T}er(\Sigma_\perp, V)$ 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 $\tau(t)|_p \neq \perp$.

Base Case. In this case there are no positions p such that $s|_p = \perp$ and $\tau(t)|_p \neq \perp$. Hence, $s = \tau(t)$ and the result is immediate when we define $s' = t$ and $\sigma' = \tau$.

Induction Step. Suppose the result holds for some number of positions $n \geqslant 0$. Let us prove the result 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. $p \in \mathcal{P}os(t)$ and $t|_p \notin V$, or
2. there exists a position $q \in \mathcal{P}os(t)$ with $t|_q \in V$ and $p = q \cdot r$.

In the first case, define:

$$t' = t[\perp]_p$$
$$\tau'(x) = \tau(x) \text{ for all } x \in V$$

In the second case, define:

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

In both cases we have $t' \preccurlyeq t$, $\tau' \preccurlyeq \tau$ and t' linear. Hence, $s \preccurlyeq \tau'(t') \prec \tau(t)$ and p is the *only* position such that $\tau'(t')|_p = \perp$ and $\tau(t)|_p \neq \perp$. Consequently, the

number of positions p with $s|_p = \bot$ and $\tau'(t')|_p \neq \bot$ is equal to n, and by the induction hypothesis there exist s' and σ' such that $s = \sigma'(s')$, $s' \preccurlyeq t'$, $\sigma' \preccurlyeq \tau'$, and s' linear. The result now follows by transitivity of the prefix orders on terms and substitutions (see Section 3.2.1). □

Two remarks are in order regarding the previous lemma:

Remark 5.1.3. If the position p as employed in the induction step is a position of a variable of t, then there is in fact more than one way to construct t' and τ'. Consider, e.g., $s = f(\bot, a)$, $t = f(x, y)$, 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 5.1.4. If t is not assumed to be linear, then the lemma does not hold. Consider, e.g., $s = f(g(\bot), g(a))$, $t = f(x, x)$, 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 $\sigma'(f(x, x)) = f(g(\bot), g(a))$, since the first argument of s is not equal to the second argument.

In some circumstances it is still possible to define s' and σ' even though t is not linear. Consider, e.g., $s = f(\bot, a)$, $t = f(x, x)$, and $\tau = [x := a]$. In this case, we have:

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

Hence, we can choose $s' = f(\bot, x)$ and $\sigma' = [x := a]$. Moreover, like in the linear case, is it sometimes even possible to define more than one s' and σ'. Consider, e.g., $s = f(\bot, \bot)$, $t = f(x, x)$, and $\tau = [x := a]$. We have:

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

and:

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

Thus, we can choose either $s' = f(x, x)$ and $\sigma' = [x := \bot]$ or $s' = f(\bot, \bot)$ and $\sigma' = [x := a]$.

5.2 Böhm-Like Trees

In this section, we define Böhm-like trees for TRSs. As a guide, we employ the three-step pattern followed in the definition of the three Böhm-like trees of the $\lambda\beta$-calculus (see Section 4.2).

As we will see, the definition of Böhm-like trees for arbitrary left-linear TRSs is slightly complicated by the fact that confluence does not necessarily hold. Therefore, we start out in Section 5.2.1 by defining Böhm-like trees for *confluent*, left-linear TRSs. In Section 5.2.2, we define Böhm-like trees for TRSs that are not necessarily confluent.

5.2.1 Confluent Systems

Following the three-step pattern of Section 4.2, we first need to define a direct approximant function from partial terms to partial terms. However, observing that more than one direct approximant function can be defined for the $\lambda\beta$-calculus and that the same may hold for TRSs, it seems inappropriate to define just a *single* direct approximant function. Hence, we define a class of direct approximant functions.

We require the value assigned to a term by a direct approximant function to represent part of the *root-stable prefix* of the term, like in the case of the $\lambda\beta$-calculus. This allows for the second and third step of the three-step pattern to be copied verbatim. Moreover, it results in Böhm-like trees that partially represent the *root-stable* part of a term as created along maximal fair reductions.

Given the explanation regarding the three properties stated at the beginning of Section 4.3.1, we have for any map from $\Lambda_\bot$ to $\Lambda_\bot$ which satisfies the three properties that the map has the root-stable prefix property we would also like to obtain in the case of TRSs. Hence, we generalise the three properties as to obtain our class of direct approximant functions:

Definition 5.2.1. *Let $\mathcal{R} = (\Sigma, R)$ be confluent and left-linear. A* direct approximant function *for $\mathcal{R}$ is a map $\omega : \mathcal{T}er(\Sigma_\bot, V) \to \mathcal{T}er(\Sigma_\bot, V)$, such that for all $s, t \in \mathcal{T}er(\Sigma_\bot, V)$ and substitutions σ it holds that:*

1. *$\omega(s) \preccurlyeq s$,*
2. *if a redex occurs at position p in s, then $\omega(s) \preccurlyeq s[\bot]_p$, and*
3. *if $s \to t$, then $\omega(s) \preccurlyeq \omega(t)$,*

where $\omega(s)$ is called the direct approximant *of s.*

As is easy to see when comparing the above definition with the three properties stated at the beginning of Section 4.3.1, the differences are minor: β-redexes and β-reductions are replaced respectively by redexes and reductions of $\mathcal{R}$.

Remark that the first clause in the above definition follows from the second one in case s is not a normal form. Moreover, remark that any direct approximant is a normal form of $\mathcal{R}$ by left-linearity.

Assuming in the remainder of this section that $\mathcal{R} = (\Sigma, R)$ is an arbitrary confluent, left-linear TRS and that ω is a direct approximant function for $\mathcal{R}$, we immediately have the following, as intended:

Lemma 5.2.2. *If $s \in \mathcal{T}er(\Sigma_\bot, V)$, then $\omega(s)$ is a root-stable prefix of s.*

Proof. Let $s \in \mathcal{T}er(\Sigma_\bot, V)$. In case s is a normal form, the result is immediate by the definition of root-stability and the first clause of Definition 5.2.1. Otherwise,

the result is immediate by the definition of root-stability and the second and third clause of Definition 5.2.1. □

We consider three examples of direct approximant functions:

Example 5.2.3 (Trivial Direct Approximant). Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. Define the *trivial* direct approximant as $\omega_{\mathrm{T}}(s) = \perp$. As $\omega_{\mathrm{T}}(s) = \perp$ for all s, we have that the clauses of Definition 5.2.1 hold trivially.

Remark that ω_{T} is minimal with respect to the prefix order in the sense that it replaces *all* root-stable subterms by $\perp$.

Example 5.2.4 (Normal Form Direct Approximant). Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. Define the *normal form direct approximant*, denoted ω_{NF}, as follows:

$$\omega_{\mathrm{NF}}(s) = \begin{cases} s & \text{if } s \text{ is a normal form} \\ \perp & \text{otherwise} \end{cases}$$

Notice that the three clauses of Definition 5.2.1 follow trivially for this map and that $\omega_{\mathrm{T}}(s) \preccurlyeq \omega_{\mathrm{NF}}(s)$ for all s. In the case of CL, we have:

$$\begin{aligned} \omega_{\mathrm{NF}}(KI) &= KI \\ \omega_{\mathrm{NF}}(KII) &= \perp \\ \omega_{\mathrm{NF}}(K(KII)) &= \perp \end{aligned}$$

Example 5.2.5 (Berarducci-Like Direct Approximant). Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. Define the *Berarducci-like* direct approximant as the map ω_{BeL} that replaces precisely all non-root-stable subterms of s by $\perp$. By definition, the three clauses of Definition 5.2.1 hold trivially. In case of CL, we have:

$$\begin{aligned} \omega_{\mathrm{BeL}}(KI) &= KI \\ \omega_{\mathrm{BeL}}(KII) &= \perp \\ \omega_{\mathrm{BeL}}(K(KII)) &= K\perp \end{aligned}$$

Note that ω_{BeL} is maximal with respect to the prefix order in the sense that it replaces *no* root-stable subterms by $\perp$. Moreover, since root-stability is in general undecidable for TRSs, we have that ω_{BeL} is in general incomputable.

The Berarducci-like direct approximant is the obvious generalisation, to arbitrary TRSs, of the Berarducci direct approximant defined for the $\lambda\beta$-calculus, as described in Section 4.2: both replace precisely all non-root-stable subterms.

Having defined a class of direct approximant functions, we proceed with the second step of the three-step pattern employed to define Böhm-like trees:

Definition 5.2.6. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. The* auxiliary set *of s (based on ω), denoted $\mathcal{A}(s)$, is defined as:*

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

Similar to the auxiliary sets of the $\lambda\beta$-calculus, the auxiliary sets of TRSs satisfy the following property:

Lemma 5.2.7. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. The set $\mathcal{A}(s)$ is directed.*

Proof. That $\mathcal{A}(s)$ is non-empty follows by the fact that $\omega(s) \in \mathcal{A}(s)$. That for all $t_1, t_2 \in \mathcal{A}(t)$ there exist $r \in \mathcal{A}(t)$ such that $t_1 \preccurlyeq r$ and $t_2 \preccurlyeq r$ follows by the third clause of Definition 5.2.1 and the assumption that $\mathcal{R}$ is confluent. □

As in the case of the $\lambda\beta$-calculus, $\mathcal{A}(s)$ is not necessarily an infinite term. To see this, consider the CL-term I. Assuming the Berarducci-Like direct approximant function, we have $\mathcal{A}(I) = \{I\}$. This set is not an infinite CL-term, as $\perp \notin \{I\}$. This brings us to the third and final step of the three-step pattern:

Definition 5.2.8. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. The* Böhm-like tree *of s (based on ω), denoted* $\mathrm{BLT}(s)$*, is defined as:*

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

By Lemma 5.2.2 and the fact that root-stability is preserved under reduction, we have that Böhm-like trees represent the root-stable part of a term as created along reductions. That maximal *fair* reductions are considered is a consequence of the definition of auxiliary sets, which takes into account *all* reductions starting from the considered term.

Since $\mathcal{A}(s)$ is directed, we have that $\downarrow\mathcal{A}(s)$ is an ideal over $\mathcal{T}er(\Sigma_\perp, V)$. Moreover, assuming ω to be fixed, it follows by definition of auxiliary sets and downward closure that each term is associated with a *unique* ideal. Hence, BLT is a map from $\mathcal{T}er(\Sigma_\perp, V)$ to $\mathcal{T}er^\infty(\Sigma_\perp, V)$.

We give three examples of Böhm-like trees:

Example 5.2.9 (Trivial Trees). Given the trivial direct approximant ω_{T} from Example 5.2.3, we can define the *trivial trees*, denoted $\mathrm{BLT}_{\mathrm{T}}$, as the Böhm-like trees based on ω_T. We have for all $s, t \in \mathcal{T}er(\Sigma_\perp, V)$ and $s \to^* t$ that $\omega_{\mathrm{T}}(t) = \perp$. Hence, $\mathcal{A}(s) = \{\perp\}$ and $\mathrm{BLT}_{\mathrm{T}}(s) = \downarrow\mathcal{A}(s) = \{\perp\}$.

Remark that trivial trees are minimal with respect to the prefix order on $\mathcal{T}er^\infty(\Sigma_\perp, V)$ in the sense that a trivial tree is always $\{\perp\}$.

Example 5.2.10 (Normal Form Trees). Given the normal form direct approximant ω_{NF} from Example 5.2.4, we can define the *normal form trees*, denoted $\mathrm{BLT}_{\mathrm{NF}}$, as the Böhm-like trees based on ω_{NF}.

The following are normal form trees for CL:

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

Here we assume Y is a fixed-point combinator, for example Curry's *paradoxical combinator* $SSI(SB(KD))$ with $D = SII$ and $B = S(KS)K$ (see, e.g., the book by Terese [Ter03]). Note that $SII(SII)$ does not have a normal form.

Example 5.2.11 (Berarducci-Like Trees). Given the Berarducci-like direct approximant ω_{BeL} from Example 5.2.5, we can define the *Berarducci-like trees*, denoted $\mathrm{BLT}_{\mathrm{BeL}}$, as the Böhm-like trees based on ω_{BeL}.

The following are Berarducci-like trees for CL:

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

We have for every $SII(SII) \to^* s$ that $s \to^* SII(SII)$. Hence, no reduct of $SII(SII)$ is root-stable.

We end this section with a proof of preservation of Böhm-like trees under rewriting. That is, we prove that Böhm-like trees for confluent, left-linear TRSs satisfy the first of the properties mentioned in Section 4.3.2 regarding the Böhm-like trees of the $\lambda\beta$-calculus:

Theorem 5.2.12. *Let* $s, t \in \mathcal{T}er(\Sigma_\bot, V)$. *If* $s \to^* t$, *then* $\mathrm{BLT}(s) = \mathrm{BLT}(t)$.

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

By definition of Böhm-like trees there exists for every $s' \in \mathrm{BLT}(s)$ a term t' such that $s \to^* t'$ and $s' \preccurlyeq \omega(t')$. Moreover, as $\mathcal{R}$ is assumed to be confluent, there exist r such that $t' \to^* r \,{}^*\!\leftarrow t$. By definition, $\omega(r) \in \mathcal{A}(t) \subseteq \mathrm{BLT}(t)$ and, by the third clause of Definition 5.2.1, $\omega(t') \preccurlyeq \omega(r)$. Hence, by transitivity of the prefix order we have $s' \preccurlyeq \omega(r)$ and $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$.

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

5.2.2 Arbitrary Systems

We next define Böhm-like trees for left-linear TRSs that are not necessarily confluent. To see that non-confluence complicates matters, consider a randomiser from a functional programming language returning either 0 or 1:

$$\begin{aligned}\mathit{Random} &\to 0\\ \mathit{Random} &\to 1\end{aligned}$$

Assuming there are no other rewrite rules, we obtain the following auxiliary set, assuming the Berarducci-like direct approximant ω_{BeL} from the previous section:

$$\mathcal{A}(\mathit{Random}) = \{0, 1\}\,.$$

Unfortunately, this auxiliary set is not directed. Hence, closing it downward does not yield an infinite term.

Assuming we want Böhm-like trees to be defined by means of downward closure, there are at least three ways to approach the above problem, given some arbitrary auxiliary set:

1. Define the Böhm-like tree as the set containing the downward closure of *each* maximal and directed subset of the auxiliary set, where maximal means that adding one more element from the auxiliary set yields a set that is no longer directed.
2. Define the Böhm-like tree as the downward closure of *one* of the maximal and directed subsets of the auxiliary set.
3. Strengthen the definition a direct approximant function such that the auxiliary set is directed even in case of a non-confluent TRS.

The first approach yields the infinite terms $\{\bot, 0\}$ and $\{\bot, 1\}$, in the case of the above example. Although this implies that every element of an auxiliary set occurs in the Böhm-like tree, the approach also has a disadvantage: Böhm-like trees are no longer infinite terms, instead the trees are sets of infinite terms. Hence, the approach requires us to reformulate the properties that hold for the Böhm-like trees of the $\lambda\beta$-calculus if we want to show that they hold with respect to a certain Böhm-like tree for TRSs.

The second approach yields either the Böhm-like tree $\{\bot, 0\}$ or the tree $\{\bot, 1\}$, in the case of the above example. Hence, given a maximal fair reduction, this approach has the disadvantage that the root-stable part as created by the reduction may be completely unrelated to the Böhm-like tree. For example, the maximal fair reduction may be $\mathit{Random} \to 0$, while the Böhm-like tree is $\{\bot, 1\}$.

The third approach implies that not every root-stable prefix may occur as the direct approximant of a term. This implies, in the case of the above example, that we obtain either $\{\bot, 0\}$, $\{\bot, 1\}$, or $\{\bot\}$, where the first two sets suffer from the same disadvantage as the second approach. The infinite term $\{\bot\}$ represents a more 'uniform' approach in the sense that the chosen maximal fair reduction now becomes irrelevant. Of course, this approach has major disadvantage: The root-stable part that is represented has become smaller.

We prefer a Böhm-like tree that consists of a single infinite term and that does not behave counterintuitively with respect to maximal fair reductions. Hence, we prefer the third of the above approaches. More in particular, we prefer the approach exemplified by $\{\bot\}$.

The set $\{\bot\}$ expresses the following principle: If a term reduces to a number of terms without a common reduct, then the direct approximant of the term and all terms it reduces to is equal to $\bot$. This principle is formalised in the fourth clause of the next definition, where it is taken into account that non-confluence is a property not only exhibited by terms but also by subterms.

Definition 5.2.13. *Let $\mathcal{R} = (\Sigma, R)$ be left-linear. A* direct approximant function *for $\mathcal{R}$ is a map $\omega : \mathcal{T}er(\Sigma_\bot, V) \to \mathcal{T}er(\Sigma_\bot, V)$, such that for all $s, t, t' \in \mathcal{T}er(\Sigma_\bot, V)$ and substitutions σ it holds that:*

1. *$\omega(s) \preccurlyeq s$,*
2. *if a redex occurs at position p in s, then $\omega(s) \preccurlyeq s[\bot]_p$,*
3. *if $s \to t$, then $\omega(s) \preccurlyeq \omega(t)$, and*
4. *if $t \;{}^*\!\leftarrow\; s \to^* t'$, then there exist $s' \in \mathcal{T}er(\Sigma_\bot, V)$ such that $t' \to^* s'$ and $\omega(t) \preccurlyeq \omega(s')$ (see Figure 5.1).*

The above definition is identical to Definition 5.2.1, except for the fourth clause, which is added to cope with non-confluence in the way described above.

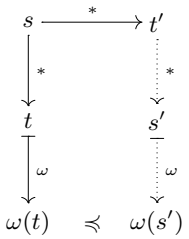

Figure 5.1. Definition 5.2.13.(4)

Definition 5.2.13 is implied by Definition 5.2.1 in case of confluent TRSs:

Proposition 5.2.14. *Let $\mathcal{R}$ be a confluent, left-linear TRS. If ω is a direct approximant function for $\mathcal{R}$ in the sense of Definition 5.2.1, then ω is also a direct approximant in the sense of Definition 5.2.13.*

Proof. Suppose ω is a direct approximant function for $\mathcal{R}$ in the sense of Definition 5.2.1. As the first three clauses of Definitions 5.2.1 and 5.2.13 correspond, we only need to prove that the fourth clause of Definition 5.2.13 is implied by the first three in the case of a confluent TRS.

Suppose $s, t, t' \in \mathcal{T}er(\Sigma_\perp, V)$. We prove that $t \,{}^*\!\leftarrow s \rightarrow^* t'$ implies there exist $s' \in \mathcal{T}er(\Sigma_\perp, V)$ such that $t' \rightarrow^* s'$ and $\omega(t) \preccurlyeq \omega(s')$. Obviously, by confluence we have that there exist r such that $t \rightarrow^* r \,{}^*\!\leftarrow t'$. By the third clause of Definition 5.2.1 it follows that $\omega(t) \preccurlyeq \omega(r)$. Hence, the result is obtained by defining $s' = r$. □

Assuming in the remainder of this section that $\mathcal{R} = (\Sigma, R)$ is a left-linear TRS and that ω is a direct approximant function for $\mathcal{R}$ in the sense of Definition 5.2.13, we prove the analogue of Lemma 5.2.2:

Lemma 5.2.15. *If $s \in \mathcal{T}er(\Sigma_\perp, V)$, then $\omega(s)$ is a root-stable prefix of s.*

Proof. Identical to the proof of Lemma 5.2.2, since the proof does not depend on the confluence of $\mathcal{R}$. □

We can also formalise the intuition behind the fourth clause:

Lemma 5.2.16. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\perp, V)$ and $p \in \mathcal{P}os(t) \cap \mathcal{P}os(t')$. If $t \,{}^*\!\leftarrow s \rightarrow^* t'$ with $t|_p$ and $t'|_p$ root-stable and $root(t|_p) \neq root(t'|_p)$, then there exist $q \in \mathcal{P}os(\omega(t))$ and $q' \in \mathcal{P}os(\omega(t'))$ such that $q, q' \leqslant p$ and $\omega(t)|_q = \omega(t')|_{q'} = \perp$.*

Proof. Let $t \,{}^*\!\leftarrow s \rightarrow^* t'$. We only prove there exist $q \in \mathcal{P}os(\omega(t))$ such that $q \leqslant p$ and $\omega(t)|_q = \perp$. The other part of the proof is completely symmetric.

By the fourth clause of Definition 5.2.13 there exist $s' \in \mathcal{T}er(\Sigma_\perp, V)$ such that $t' \rightarrow^* s'$ and $\omega(t) \preccurlyeq \omega(s')$. Moreover, as $t'|_p$ is root-stable, we have $p \in \mathcal{P}os(s')$,

$s'|_p$ root-stable, and $root(s'|_p) = root(t'|_p)$. Hence, $root(s'|_p) \neq root(t|_p)$ and, by the fact that $\omega(t) \preccurlyeq \omega(s')$, there exists a position $q \in \mathcal{P}os(\omega(t)) \cap \mathcal{P}os(\omega(s')) \subseteq \mathcal{P}os(\omega(t))$ such that $q \leqslant p$ and $\omega(t)|_q = \bot$. □

For a further discussion of the fourth clause of Definition 5.2.13, in combination with related work, see Section 5.5.

An example of a direct approximant function that satisfies Definition 5.2.13 is the following:

Example 5.2.17 (Trivial Direct Approximant). The trivial direct approximant, as defined in Example 5.2.3 is a direct approximant function in the sense of Definition 5.2.13, since $\omega_{\mathrm{T}}(s) = \bot$ for all terms $s \in \mathcal{T}er(\Sigma_\bot, V)$.

It is readily proved that the normal form direct approximant and the Berarducci-like direct approximant do not satisfy Definition 5.2.13 in case a non-confluent TRS is considered. However, by Lemma 5.2.14 the definition is satisfied in case of a confluent TRS.

Having defined direct approximant functions for arbitrary left-linear TRSs, we pick up the three-step pattern employed in Section 4.2 to define the Böhm-like trees of the $\lambda\beta$-calculus and we define auxiliary sets:

Definition 5.2.18. *Let $s \in \mathcal{T}er(\Sigma_\bot, V)$. The* auxiliary set *of s (based on ω), denoted $\mathcal{A}(t)$, is defined as:*

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

As in the case of confluent systems, but assuming the new definition of a direct approximant function, we have the following:

Lemma 5.2.19. *Let $s \in \mathcal{T}er(\Sigma_\bot, V)$. The set $\mathcal{A}(s)$ is directed.*

Proof. That $\mathcal{A}(s)$ is non-empty follows by the fact that $\omega(s) \in \mathcal{A}(s)$. That for all $t_1, t_2 \in \mathcal{A}(s)$ there exist $r \in \mathcal{A}(s)$ such that $t_1 \preccurlyeq r$ and $t_2 \preccurlyeq r$ is immediate by the third and fourth clause of Definition 5.2.13. □

As before, $\mathcal{A}(s)$ is not necessarily a tree. This follows directly by Proposition 5.2.14 and the counterexample from the previous section.

Continuing with the third step of the three-step pattern, we can define:

Definition 5.2.20. *Let $s \in \mathcal{T}er(\Sigma_\bot, V)$. The* Böhm-like tree *of s (based on ω), denoted* $\mathrm{BLT}(s)$*, is defined as:*

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

By Lemmas 5.2.15 and 5.2.16 and the preservation of root-stability under reduction, it follows that a Böhm-like tree can only represent root-stable subterms that do not differ between different reductions. Moreover, maximal fair reductions are considered by definition of the auxiliary sets.

As in the case of the previous section, we have that each Böhm-like tree based on a particular direct approximant function associates a unique infinite term with every term. Hence, BLT is a map from $\mathcal{T}er(\Sigma_\bot, V)$ to $\mathcal{T}er^\infty(\Sigma_\bot, V)$.

As before, Böhm-like trees are preserved under rewriting:

Theorem 5.2.21. *Let $s, t \in \mathcal{T}er(\Sigma, V)$. If $s \to^* t$, then* $\mathrm{BLT}(s) = \mathrm{BLT}(t)$.

Proof. Suppose $s \to^* t$. We prove $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$ and $\mathrm{BLT}(t) \preccurlyeq \mathrm{BLT}(s)$. The result is then immediate by the observation that the prefix order is actually subset inclusion.

By the definition of Böhm-like trees there exists for every $s' \in \mathrm{BLT}(s)$ a term t' such that $s \to^* t'$ and $s' \preccurlyeq \omega(t')$. Moreover, by the fourth clause of Definition 5.2.13 there exists a term s'' such that $t \to^* s''$ and $\omega(t') \preccurlyeq \omega(s'')$. By $t \to^* s''$ and the definition of auxiliary sets we have that $\omega(s'') \in \mathcal{A}(t) \subseteq \mathrm{BLT}(t)$. Hence, by $\omega(t') \preccurlyeq \omega(s'')$ and transitivity of the prefix order we have that $s' \preccurlyeq \omega(s'')$ 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 definition of downward closure $\downarrow\mathcal{A}(t) \subseteq \downarrow\mathcal{A}(s)$. Thus, $\mathrm{BLT}(t) \preccurlyeq \mathrm{BLT}(s)$. □

5.3 Monotonicity and Continuity

As mentioned in Section 4.3.2, monotonicity and continuity hold for the Böhm and Lévy-Longo trees due to the fact the direct approximant functions of these trees satisfy the additional property of being monotone. That is, the functions satisfy that $s \preccurlyeq t$ implies $\omega(s) \preccurlyeq \omega(t)$. We next show that the same holds with respect to Böhm-like trees for TRSs.

Assume that $\mathcal{R} = (\Sigma, V)$ is a left-linear TRS and that ω is a direct approximant function for $\mathcal{R}$ that is *monotone*. We can prove the following lemma and theorem:

Lemma 5.3.1. *The Böhm-like tree based on ω is monotone. That is, for all $s, t \in \mathcal{T}er(\Sigma_\perp, V)$, if $s \preccurlyeq t$, then* $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$.

Proof. Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$ such that $s \preccurlyeq t$. Suppose $s'' \in \mathrm{BLT}(s)$. By the definition of $\mathrm{BLT}(s)$ there exist s' such that $s'' \preccurlyeq \omega(s')$ and $s \to^* s'$. Moreover, by left-linearity of $\mathcal{R}$ there exist t' such that $t \to^* t'$ and $s' \preccurlyeq t'$. Hence, since ω is assumed to be monotone, we have $\omega(s') \preccurlyeq \omega(t')$. Thus, as $\omega(t') \in \mathrm{BLT}(t)$, we have also $s'' \in \mathrm{BLT}(t)$ and $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$. □

Theorem 5.3.2. *The Böhm-like tree based on ω is continuous. That is, if $s \in \mathcal{T}er(\Sigma_\perp, V)$, then* $\mathrm{BLT}(s) = \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq s\}$.

Proof. Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. Because $s \preccurlyeq s$, we have $\mathrm{BLT}(s) \in \{\mathrm{BLT}(t) \mid t \preccurlyeq s\}$. Hence, $\mathrm{BLT}(s) \preccurlyeq \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq s\}$. Moreover, by Lemma 5.3.1 we have for all $t \preccurlyeq s$ that $\mathrm{BLT}(t) \preccurlyeq \mathrm{BLT}(s)$. Thus, $\bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq s\} \preccurlyeq \mathrm{BLT}(s)$. Combining both facts yields the desired result. □

The following holds irrespective of any additional assumptions on ω:

Proposition 5.3.3. *If a Böhm-like tree is continuous, then it is monotone.*

Proof. Suppose $s, t \in \mathcal{T}er(\Sigma_\perp, V)$ and $s \preccurlyeq t$. Obviously, we have:

$$\mathrm{BLT}(s) \in \{\mathrm{BLT}(t') \mid t' \preccurlyeq t\}\,.$$

Hence, by continuity:

$$\mathrm{BLT}(s) \preccurlyeq \bigsqcup\{\mathrm{BLT}(t') \mid t' \preccurlyeq t\} = \mathrm{BLT}(t)\,,$$

as required. □

We next consider the monotonicity and continuity of the Böhm-like trees for TRSs which has so far occurred in this chapter:

Example 5.3.4. With respect to the trivial direct approximant it holds for all $s, t \in \mathcal{T}er(\Sigma_\bot, V)$ that $\omega_\mathrm{T}(s) = \omega_\mathrm{T}(t) = \bot$. Hence, $s \preccurlyeq t$ implies $\omega_\mathrm{T}(s) = \omega_\mathrm{T}(t) = \bot$, which means that the trivial direct approximant is monotone and that the trivial trees are monotone and continuous.

Example 5.3.5. Assuming $\mathcal{R}$ is confluent, it generally not the case that the normal form trees from Example 5.2.10 are monotone and continuous. To see this, consider the CL-terms $\bot I$ and II. The term $\bot I$ is a normal form, $II \to I$, and $\bot I \preccurlyeq II$. We have:

$$\mathrm{BLT}_\mathrm{NF}(\bot I) = \{\bot, \bot\bot, \bot I\} \not\preccurlyeq \{\bot, I\} = \mathrm{BLT}_\mathrm{NF}(II)$$

and

$$\mathrm{BLT}_\mathrm{NF}(II) = \{\bot, I\} \neq \{\bot, \bot, \bot, \bot I, I\} = \bigcup\{\mathrm{BLT}_\mathrm{NF}(s) \mid s \preccurlyeq II\}\,.$$

Hence, normal form trees are neither monotone nor continuous (remember that the least upper bound of a set of infinite terms is their union). In fact, the set $\bigcup\{\mathrm{BLT}_\mathrm{NF}(s) \mid s \preccurlyeq II\}$ is not even an infinite term.

Example 5.3.6. Assuming once again that $\mathcal{R}$ is confluent, it generally does not hold either that the Berarducci-like trees from Example 5.2.11 are monotone and continuous. The counterexample is identical to the counterexample provided above in the case of normal form trees.

Although we proved above that:

$$\mathrm{BLT}(s) = \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq s\}\,, \tag{5.1}$$

it is more common to prove for every context $C[\Box]$ that:

$$\mathrm{BLT}(C[s]) = \bigsqcup\{\mathrm{BLT}(C[t]) \mid t \preccurlyeq s\}\,. \tag{5.2}$$

However, these two statements are equivalent:

Proposition 5.3.7. *Equation* (5.1) *holds if and only if Equation* (5.2) *holds.*

Proof. Equation (5.1) follows directly from Equation (5.2) by substitution of the context $\Box$ for $C[\Box]$.

To show the reverse, consider the following instantiation of Equation (5.1):

$$\mathrm{BLT}(C[s]) = \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq C[s]\}\,.$$

For every t in Equation (5.2) we have $C[t] \preccurlyeq C[s]$. Hence,

$$\bigsqcup\{\mathrm{BLT}(C[t]) \mid t \preccurlyeq s\} \subseteq \bigsqcup\{\mathrm{BLT}(t) \mid s \preccurlyeq C[s]\}$$

Moreover, for every $t \preccurlyeq C[s]$, as occurring in the above instantiation of Equation (5.1), there exist $C[s'] \preccurlyeq C[s]$ such that $t \preccurlyeq C[s']$. From this, and Proposition 5.3.3, it follows that:

$$\bigsqcup\{\mathrm{BLT}(C[t]) \mid t \preccurlyeq s\} \supseteq \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq C[s]\}$$

Hence,

$$\bigsqcup\{\mathrm{BLT}(C[t]) \mid t \preccurlyeq s\} = \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq C[s]\}\,,$$

as required. □

5.4 Direct Approximant TRSs

As observed in Section 4.2, it is possible to define terminating rewrite systems with the property that each λ-term has a unique normal form which is either the Böhm or Lévy-Longo direct approximant of the term, depending on the particular rewrite system.

In this section we imitate the rewriting approach in the context of TRSs. To this end, we define a class of terminating TRSs, the *direct approximant TRSs (ωTRSs)*, all whose members have the property that each term has a unique normal form. We show that the map that assigns to each term its unique normal form is a direct approximant function.

As an added bonus, we will obtain that each of the defined direct approximant functions is monotone with respect to the prefix order. Hence, a Böhm-like tree is monotone and continuous whenever it is based on a direct approximant function definable by means of an ωTRS.

Remark that the Berarducci-like direct approximant cannot be defined by means of a finite, terminating TRS with unique normal forms: Such a TRS implies that the direct approximants are computable, while the Berarducci-like direct approximants are incomputable by undecidability of root-stability.

Limiting ourselves to confluent, left-linear TRSs, as in the case of Section 5.2.1, the class of TRSs is defined as follows:

Definition 5.4.1. *Let $\mathcal{R} = (\Sigma, R)$ be a confluent, left-linear TRS. A* direct approximant TRS (ωTRS) *for $\mathcal{R}$ 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. *$s \to_\omega^* \perp$ for all $s \preccurlyeq d$ with $d \to_\omega \perp \in D$ (see Fig. 5.2), and*
4. *$l \to_\omega^* \perp$ for all $l \to r \in R$.*

The above definition almost copies verbatim the properties shared between the rewrite systems of the Böhm and Lévy-Longo direct approximants. There are *two* differences: In the third and fourth clause $\to_\omega^*$ is employed instead of respectively $\to_\omega^=$ and $\to_\omega$. The reason for these differences is explained below.

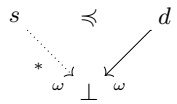

Figure 5.2. Definition 5.4.1.(3)

In the remainder of this section we assume that $\mathcal{R} = (\Sigma, R)$ is a confluent, left-linear TRS. Moreover, we assume that $\mathcal{D} = (\Sigma_\bot, D)$ is an ωTRS for $\mathcal{R}$.

We proceed as follows: We first give an example of an ωTRS and we explain the reason for the occurrence of $\to_\omega^*$ in Definition 5.4.1. Thereafter, we prove that ωTRSs are confluent and terminating, employing the first, second, and third clause of Definition 5.4.1. Finally, employing the third and fourth clause of the definition, we show that the unique normal forms define a monotone direct approximant function.

Example 5.4.2 (Huet-Lévy ωTRS). The *Huet-Lévy ωTRS* is defined as $\mathcal{HL} = (\Sigma_\bot, HL)$, where $d \to_\omega e \in HL$ if and only if $e = \bot$ and $d \preccurlyeq l$ for some $l \to r \in R$. That $\mathcal{HL}$ is an ωTRS, as its name suggests, follows readily from the definition.

The Huet-Lévy ωTRS occurs first in the work by Klop and Middeldorp [KM91]. The unique normal forms of the TRS define a map which was first formulated by Huet and Lévy [HL91]. The definition of Klop and Middeldorp differs slightly from ours, but equality of the transitive-reflexive closures follows easily with the help of Lemma 5.1.2.

The Huet-Lévy ωTRS for CL has no less than *28 rewrite rules*. However, employing the fact that $\to_\omega^*$ occurs in the third and fourth clause of Definition 5.4.1, we can define an ωTRS which the same transitive-reflexive closure as the Huet-Lévy ωTRS, but which only has *four* rewrite rules:

$$\begin{aligned} Sxyz &\to_\omega \bot & \qquad Kxy &\to_\omega \bot \\ Ix &\to_\omega \bot & \qquad \bot x &\to_\omega \bot \end{aligned}$$

Hence, the formulation of the third and fourth clause of Definition 5.4.1 allow for some *economy of size* in ωTRSs.

To prove confluence of the assumed ωTRS $\mathcal{D}$, we first show that confluence holds for ωTRSs that allow the third clause of Definition 5.4.1 to be strengthened to:

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

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

Proposition 5.4.3. *If $\mathcal{E} = (\Sigma_\perp, E)$ is a single-step ωTRS, then $\mathcal{E}$ is confluent.*

Proof. Given a single-step ωTRS $\mathcal{E} = (\Sigma_\perp, E)$, we prove that $\mathcal{E}$ is subcommutative. Confluence is implied by subcommutativity, as remarked in Section 2.2.2.

Let $s, t_1, t_2 \in \mathcal{T}er(\Sigma_\perp, V)$ and suppose $t_1 \ {}_\omega\!\leftarrow s \rightarrow_\omega t_2$. Assume the contracted redexes occur respectively at the positions p_1 and p_2. Without loss of generality, there are three cases to consider depending on the relative positions of p_1 and p_2:

The positions p_1 and p_2 are parallel. In this case, a redex occurs at position p_1 in t_2 and one also occurs at position p_2 in t_1. By the first clause of Definition 5.4.1 contracting both redexes results in the same term, which completes this case.

The positions p_1 and p_2 are equal. In this case, we are done immediately, again by the first clause of Definition 5.4.1.

The position p_1 is a prefix of p_2. By the first clause of Definition 5.4.1 we have $t_2|_{p_1} \preccurlyeq s|_{p_1}$ and $t_1|_{p_1} = \perp$. Moreover, as $s|_{p_1} \rightarrow_\omega t_1|_{p_1} = \perp$, we have by left-linearity of $\mathcal{E}$, Lemma 5.1.2, and the single-step assumption that $t_2|_{p_1} \rightarrow_\omega^= \perp = t_1|_{p_1}$. Hence, $t_2 \rightarrow_\omega^= t_1$, which completes this last case. □

Employing confluence of single-step ωTRSs we can prove confluence of $\mathcal{D}$:

Lemma 5.4.4. *The ωTRS $\mathcal{D}$ is confluent.*

Proof. Define a TRS $\mathcal{E} = (\Sigma_\perp, E)$ such that $s \rightarrow_\omega \perp \in E$ for all $s \in \mathcal{T}er(\Sigma_\perp, V)$ with $\perp \neq s \preccurlyeq d$ and $d \rightarrow_\omega \perp \in D$. By definition, the TRS $\mathcal{E}$ is a single-step ωTRS whose transitive-reflexive closure is equal to that of $\mathcal{D}$. Hence, confluence of $\mathcal{D}$ follows by Proposition 5.4.3. □

To prove termination of $\mathcal{D}$ we first prove the following with respect to the rewrite relation of $\mathcal{D}$:

Proposition 5.4.5. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$. If $s \rightarrow_\omega t$, then $s \succ t$.*

Proof. By the first clause of Definition 5.4.1 we have that $s \rightarrow_\omega t$ is a replacement of a subterm $s|_p$ by $\perp$. Since $s|_p \succ \perp$ by the second clause of Definition 5.4.1, it follows that $s = s[s|_p]_p \succ s[\perp]_p = t$, as required. □

We can now prove termination:

Lemma 5.4.6. *The ωTRS $\mathcal{D}$ is terminating.*

Proof. Immediate by Propositions 5.4.5 and 3.2.5. □

By Lemmas 5.4.4 and 5.4.6, we have that each term $s \in \mathcal{T}er(\Sigma_\perp, V)$ has a unique normal form with respect to $\mathcal{D}$. We denote the unique normal form of a term s by $\omega(s)$.

We next prove that ω defines a monotone direct approximant function. To facilitate the proof, we first establish a number of facts relating ωTRSs with the prefix order terms. This is the contents of the following three lemmas.

Lemma 5.4.7. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\perp, V)$. If $s \preccurlyeq t$ and $t \rightarrow_\omega^* t'$, then there exist $s' \in \mathcal{T}er(\Sigma_\perp, V)$ such that $s' \preccurlyeq t'$ and $s \rightarrow_\omega^* s'$ (see Figure 5.3).*

Proof. We give a proof for $t \to_\omega t'$. The result then follows by induction on the length of $t \to_\omega^* t'$. Thus, suppose that $s \preccurlyeq t$ and that 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 position $q \leqslant p$ such that $s|_q = \bot$. Define $s' = s$. As $t \to_\omega t'$ replaces the subterm at position p by $\bot$, we have by $s|_q = \bot$ and $q \leqslant p$ that $s \preccurlyeq t'$. Moreover, it is immediate that $s \to_\omega^* s = s'$.

The position p occurs in s. In this case, $s|_p \preccurlyeq t|_p$. As $t|_p$ is a redex, we have by Lemma 5.1.2 and the third clause of Definition 5.4.1 that $s|_p \to_\omega^* \bot = t'|_p$. Define $s' = s[\bot]_p$. Since $t' = t[\bot]_p$, it follows that $s' \preccurlyeq t'$. Moreover, as $s|_p \to_\omega^* \bot$, we have $s \to_\omega^* s'$. □

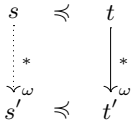

Figure 5.3. Lemma 5.4.7

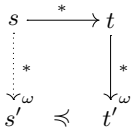

Figure 5.4. Lemma 5.4.8

Lemma 5.4.8. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\bot, V)$. If $s \to^* t$ and $t \to_\omega^* t'$, then there exist $s' \in \mathcal{T}er(\Sigma_\bot, V)$ such that $s \to_\omega^* s'$ and $s' \preccurlyeq t'$ (see Figure 5.4).*

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

Suppose the redex contracted in $s \to t$ occurs at position p. As $s[\bot]_p \preccurlyeq t$, there exists by Lemma 5.4.7 a term s' such that $s' \preccurlyeq t'$ and $s[\bot]_p \to_\omega^* s'$. Moreover, $s \to_\omega^* s'$, because, by the fourth clause of Definition 5.4.1, we have that $s \to_\omega^* s[\bot]_p$. □

Lemma 5.4.9. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$. The following properties hold:*

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

Proof. Recall that $\omega(s)$ is the unique normal form of s with respect to $\mathcal{D}$. We prove each of the five clauses in turn:

1. Since $\omega(s)$ is the unique normal form of s, we have $s \to_\omega^* \omega(s)$. The result follows by repeated application of Proposition 5.4.5.
2. For every $s|_p \to_\omega^* t$ we have $s = s[s|_p]_p \to_\omega^* s[t]_p$. Hence, as $s|_p \to_\omega^* \omega(s|_p)$, 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 5.4.7 an s' such that $s' \preccurlyeq \omega(t)$. By confluence of ωTRSs and the first clause of the current lemma we have $\omega(s) = \omega(s') \preccurlyeq s'$. Hence, by transitivity of the prefix order $\omega(s) \preccurlyeq \omega(t)$.
5. Analogous to the previous clause of the current lemma employing Lemma 5.4.8 instead of Lemma 5.4.7. □

We can now prove the main theorem of this section:

Theorem 5.4.10. *The map $\omega : \mathcal{T}er(\Sigma_\perp, V) \to \mathcal{T}er(\Sigma_\perp, V)$ which assigns to each term its unique normal form with respect to $\mathcal{D}$ is a monotone direct approximant function.*

Proof. Since we assume $\mathcal{R}$ is confluent, we can show that ω is a direct approximant function by verifying the three clauses of Definition 5.2.1: The first clause is immediate by Lemma 5.4.9.(1); the second clause follows by the fourth clause of Definition 5.4.1 and Lemma 5.4.9.(2); the third clause follows by Lemma 5.4.9.(5). That ω is monotone follows by Lemma 5.4.9.(4). □

By the previous theorem, we have that each ωTRS defines a Böhm-like tree that is monotone and continuous. In particular, we have:

Example 5.4.11 (Huet-Lévy Trees). The *Huet-Lévy tree*, denoted $\mathrm{BLT_{HL}}$ is the tree based on the Huet-Lévy ωTRS of Definition 5.4.2. The Huet-Lévy tree has previously been defined by Boudol [Bou85] and Ariola [Ari96].

It is readily proved for all $s \in \mathcal{T}er(\Sigma_\perp, V)$ that $\mathrm{BLT_T}(s) \preccurlyeq \mathrm{BLT_{HL}}(s)$ and that $\mathrm{BLT_{HL}}(s) \preccurlyeq \mathrm{BLT_{BeL}}(s)$. For example, in case of the CL-term $s = K(SII(SII)I)$, we have:

$$\begin{aligned}\mathrm{BLT_T}(s) &= \downarrow\{\perp\}\\ \mathrm{BLT_{HL}}(s) &= \downarrow\{K\perp\}\\ \mathrm{BLT_{BeL}}(s) &= \downarrow\{K(\perp I)\}\end{aligned}$$

In the case of normal form trees, we have neither $\mathrm{BLT_{NF}}(s) \preccurlyeq \mathrm{BLT_{HL}}(s)$ nor $\mathrm{BLT_{HL}}(s) \preccurlyeq \mathrm{BLT_{NF}}(s)$, as is easily seen when considering the CL-terms $s = \perp I$ and $t = K(SII(SII))$:

$$\begin{aligned}\mathrm{BLT_{NF}}(s) &= \downarrow\{\perp I\} & \mathrm{BLT_{NF}}(t) &= \downarrow\{\perp\}\\ \mathrm{BLT_{HL}}(s) &= \downarrow\{\perp\} & \mathrm{BLT_{HL}}(t) &= \downarrow\{K\perp\}\end{aligned}$$

Discussion. Assume once more that $\mathcal{R} = (\Sigma, \perp)$ is a confluent, left-linear TRS and that $\mathcal{D} = (\Sigma_\perp, D)$ is an ωTRS for $\mathcal{R}$. By the fourth clause of Definition 5.4.1 and Lemma 5.4.7, we have for every $l \to r \in R$ and $s \preccurlyeq l$ that $s \to_\omega^* \perp$. Since for every such $s \neq \perp$ the Huet-Lévy ωTRS has a rule $s \to_\omega \perp$, it holds for each $s \to_\omega^* t$ with respect to the Huet-Lévy ωTRS that there exists a reduction $s \to_\omega^* t$ with respect to $\mathcal{D}$. Hence, given any term s we have $\omega(s) \preccurlyeq \omega_{\mathrm{HL}}(s)$, where ω is the direct approximant function based on $\mathcal{D}$.

The above observation raises a question of universality: Why do we consider arbitrary ωTRSs and not just the Huet-Lévy ωTRS? The answer is motivated by the direct approximants of the $\lambda\beta$-calculus: There exist two direct approximant functions definable by means of the unique normal forms of a terminating rewrite system, i.e., the Böhm and Lévy-Longo direct approximant functions.

The rewrite system that defines the Lévy-Longo direct approximant is in some sense the Huet-Lévy direct approximant of the $\lambda\beta$-calculus: The transitive-reflexive closure of the Lévy-Longo rewrite system is identical to the transitive-reflexive closure of the rewrite system that has for each $\bot \neq s' \preccurlyeq (\lambda x.s)t$ a rule $s' \to_\omega \bot$, i.e., the rewrite system that consists of the following rewrite rules:

$$\begin{aligned} (\lambda x.s)t &\to_\omega \bot & \qquad (\lambda x.\bot)t &\to_\omega \bot \\ (\lambda x.s)\bot &\to_\omega \bot & (\lambda x.\bot)\bot &\to_\omega \bot \\ \bot t &\to_\omega \bot & \bot\bot &\to_\omega \bot \end{aligned}$$

This raises a new question: Why consider the Böhm direct approximant function?

To answer this question, consider the left- and right-hand side of the rewrite rule $\lambda x.\bot \to_\omega \bot$, which is particular for the Böhm direct approximant. Moreover, consider an arbitrary context $C[\Box]$. The terms $C[\lambda x.\bot]$ and $C[\bot]$ allow for exactly the same β-reductions, with only one exception. If $\Box$ occurs as $\Box t$, then $(\lambda x.\bot)t \to_\beta \bot$, while no β-redex occurs at the root of $\bot t$. However, the Lévy-Longo direct approximants of $\bot t$ and $\bot$ are equal. Hence, in every context either the behaviour of the terms is the same or they have the same Lévy-Longo direct approximant after a single β-reduction. So, why not omit the single β-reduction? This is exactly what is provided for by the rule $\lambda x.\bot \to_\omega \bot$.

Under the assumption of the Huet-Lévy ωTRS for CL, it is possible to observe behaviour that similar to that of the λ-terms $\lambda x.\bot$ and $\bot$, e.g., in the case of the CL-terms $S\bot$, $K\bot$, and $\bot$. This justifies the addition of the following two rewrite rules to the Huet-Lévy ωTRS:

$$\begin{aligned} S\bot &\to_\omega \bot \\ K\bot &\to_\omega \bot \end{aligned}$$

It is readily shown that the addition of the above two rewrite rules yields an ωTRS for CL which is sensible in view of the above discussion. For this reason, we consider ωTRSs in the sense of Definition 5.4.1 and not just Huet-Lévy ωTRSs.

5.5 Related Work

Work related to that presented in the current chapter can be divided into three categories: concrete Böhm-like trees for TRSs, more general definitions of Böhm-like trees employing some notion of direct approximants, and infinitary rewriting. The first two categories are discussed here. The last category is the subject of Chapter 7.

Concrete Trees. As mentioned in Example 5.4.11, Boudol [Bou85] and Ariola [Ari96] already define the Huet-Lévy tree. In both cases, the definition proceeds along the lines of the previous sections. The observation that Huet-Lévy trees are monotone and continuous, which follows immediately by the remark just below Theorem 5.4.10, already occurs in Ariola's work [Ari96].

More General Definitions. Definitions of Böhm-like trees for TRSs that are more general than the definition presented in the current chapter occur in the work by Boudol [Bou85], Blom [Blo01], and Ariola and Blom [AB02]. In essence, all three generalisations still follow the three-step pattern of the $\lambda\beta$-calculus, as discussed in Section 4.2. However, in each case some of the restrictions of the three-step pattern are relaxed. We discuss each of the generalisations in turn.

Boudol. In his work, Boudol [Bou85] generalises the definition presented in the current chapter with respect to two points: the codomain of direct approximant functions and the definition of auxiliary sets.

Boudol's definition of a direct approximant function consists of the second and third clause of Definition 5.2.13. The first and fourth clause of Definition 5.2.13 are dropped. The codomain of a direct approximant function is no longer required to be $\mathcal{T}er^\infty(\Sigma_\bot, V)$. It may be any coalgebra over $\Sigma_\bot$ which has an order defined over it such that the interpretation of $\bot$ is the least element and such that all function symbols of Σ are interpreted as continuous functions.

Although Boudol's definition of a direct approximant function is more general with respect to the codomain, it is also less general in some sense: By the continuous interpretation of the function symbols, direct approximant functions are always monotone. Hence, Berarducci-like trees cannot be defined when adopting Boudol's approach.

To overcome the removal of the fourth clause of Definition 5.2.13 and the use of non-confluent systems, Boudol employs a variant of the first approach discussed in Section 5.2.2. He defines auxiliary sets as:

$$\mathcal{A}(s) = \bigcup\nolimits_{(s_i)_{i<n} \in S} \{\omega(s_k) \mid s_k \text{ occurs in } (s_i)_{i<n}\}\,,$$

where S denotes the set of all reduction sequences over terms in $\mathcal{T}er(\Sigma_\bot, V)$. Böhm-like trees are then defined as usual:

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

Boudol's definition of Böhm-like trees no longer yields infinite terms for two reasons: The codomain of the direct approximant function does no longer need to have a term structure and the definition of the auxiliary set allows the Böhm-like tree to be something other than an ideal. Note, however, that some notion of root-stability is still present, since the second clause of Definition 5.2.13 is maintained.

Blom. In his dissertation, Blom [Blo01] defines Böhm-like trees for arbitrary ARSs. The differences between Blom's work and the current chapter all relate to the definition of the direct approximant function. Blom's definitions of auxiliary sets and Böhm-like trees are identical to those presented here.

Given an ARS $\mathcal{A} = (A, \to)$ and a CPO $\mathcal{B} = (B, \sqsubseteq)$, Blom defines a direct approximant function as a map $\omega : A \to B$ which satisfies two properties:

1. if $a \to b$, then $\omega(a) \sqsubseteq \omega(b)$, and
2. if $b \,{}^*\!\leftarrow a \to^* b'$, then there exist $a' \in A$ such that $b' \to^* a'$ and $\omega(b) \sqsubseteq \omega(a')$.

Thus, the first and the second clause of Definition 5.2.13 are dropped and the third and fourth clause are generalised appropriately. Remark that, by omitting the second clause, we no longer have that Böhm-like trees represent root-stable parts.

Blom [Blo01, Theorem 6.2.7] shows that the second clause of his definition cannot be omitted if each term is required to have a unique Böhm-like tree. As Definition 5.2.13 is an instance of Blom's definition, the same holds in our case.

The second clause of Blom's direct approximant definition is actually an instance of the following concept, as also defined by Blom [Blo01]:

Definition 5.5.1. *Let $\mathcal{A} = (A, \to_\alpha, \to_\beta)$ be an ARS. The relation $\to_\alpha$ is* skew confluent *with respect to $\to_\beta$, if for all $a \to_\alpha^* b$ and $a \to_\alpha^* c$ with $a, b, c \in A$ there exist $d \in A$ such that $b \to_\alpha^* d$ and $c \to_\beta^* d$ (see Figure 5.5).*

To see that the second clause is an instance, define $a \to_\alpha b$ as $a \to b$ and $a \to_\beta b$ as $\omega(a) \sqsubseteq \omega(b)$.

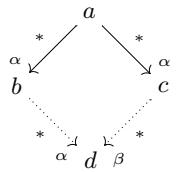

Figure 5.5. Definition 5.5.1

Ariola and Blom. The relevant parts of the work of Ariola and Blom [AB02] are essentially identical to those from Blom's dissertation [Blo01]. For this reason, we do not discuss the work of Ariola and Blom any further here.

6

Congruence

Zaphod glared at him with resentment and loathing.
"What's the difference?" he muttered.
"Nothing," said Zarniwoop, "they are identical."

— Douglas Adams
The Restaurant at the End of the Universe (1980)

Given a TRS $\mathcal{R} = (\Sigma, R)$ and some set D, we can define a *denotational semantics* for $\mathcal{R}$ to be any map ϕ from $\mathcal{T}er(\Sigma, V)$ to D. However, as for example remarked by Lévy [Lév78] and Welch [Wel75], not every map defines a 'proper' denotational semantics. A proper denotational semantics requires ϕ to satisfy at least two properties:

1. if $s \to^* t$, then $\phi(s) = \phi(t)$, and
2. if $\phi(s) = \phi(t)$ and $C[\Box]$ a context, then $\phi(C[s]) = \phi(C[t])$.

Hence, a proper denotational semantics is preserved under rewriting and it is a congruence with respect to equality. In other words, a proper denotational semantics is a model of the TRS (see, e.g., the book by Baader and Nipkow [BN98]).

Recall that Böhm-like trees for TRSs, as defined in the previous chapter, are maps from $\mathcal{T}er(\Sigma_\perp, V)$ to $\mathcal{T}er^\infty(\Sigma_\perp, V)$. Hence, since the domain is $\mathcal{T}er(\Sigma_\perp, V)$, we can ask ourselves whether Böhm-like trees define a proper denotational semantics. As such, we need to prove the two properties specified above. The first property already occurs in Chapter 5 and is shown to hold for every Böhm-like tree in Theorem 5.2.21. The second property is the subject of this chapter.

As we show in Section 6.1, Böhm-like trees based on direct approximant functions and ωTRSs are generally *not* congruent with respect to Böhm-like tree equality. However, as we show in Section 6.2, there is an easy way to strengthen the definition of direct approximant functions such that congruence is implied. Unfortunately, the strengthening is only deceptively easy: The proof obligations that exist with respect to congruence are not simplified in any way.

We can simplify the proof obligations by assuming *syntactic continuity*, which implies congruence, as we show in Section 6.3. Syntactic continuity is defined as follows, where s is a term and $C[\Box]$ a context:

$$\mathrm{BLT}(C[s]) = \bigsqcup \{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}\,.$$

Of course, syntactic continuity does not hold either in the case of the Böhm-like trees defined in Chapter 5, since congruence does *not* hold.

To overcome the above problem, we show in Section 6.4 how to strengthen the definition of ωTRSs such that syntactic continuity is implied. We do not strengthen

the definition of direct approximant functions: We still want to be able to define Berarducci-like trees. These trees do not satisfy syntactic continuity, as we show in Section 6.3.

Partially summarising the above, this chapter is arranged as follows: In Section 6.1, we explain why congruence does not hold for Böhm-like trees as defined in the previous chapter. In Section 6.2, we strengthen the definition of direct approximant functions such that congruence is implied. Thereafter, in Section 6.3, we show that syntactic continuity implies congruence. In addition, we show that syntactic continuity implies precongruence, a property slightly stronger than congruence. Finally, in Section 6.4, the definition of ωTRSs is strengthened as to obtain syntactic continuity.

As in Chapter 5, Combinatory Logic is employed in most examples presented in this chapter. There are also some examples taken from the realm of functional programming.

6.1 Congruence

Given a Böhm-like tree BLT for a left-linear TRS, congruence is defined as:

$$\mathrm{BLT}(s) = \mathrm{BLT}(t) \text{ implies } \mathrm{BLT}(C[s]) = \mathrm{BLT}(C[t])\,,$$

where s and t are terms, where $C[\Box]$ is a context. Above, BLT is substituted for the arbitrary map ϕ as employed in the introduction.

To see that congruence does not necessarily hold, even in the case of Böhm-like trees defined by means of an ωTRS, consider the following rewrite rules:

$$\begin{aligned} \mathit{IsEmpty}(\mathit{nil}) &\to \mathit{True} \\ \mathit{IsEmpty}(x : xs) &\to \mathit{False} \end{aligned}$$

The rules are taken from functional programming and can be employed to determine whether a list is empty or not. With respect to the rewrite rules, we can define the following ωTRS:

$$\begin{aligned} \mathit{IsEmpty}(xs) &\to_\omega \bot \\ \mathit{nil} &\to_\omega \bot \end{aligned}$$

Although slightly absurd, since $\mathrm{BLT}(\mathit{nil}) = \{\bot\}$ and not $\{\bot, \mathit{nil}\}$, the rules actually define an ωTRS, as is readily verified.

Now consider the terms $\bot$ and nil. We obviously have:

$$\mathrm{BLT}(\bot) = \{\bot\} = \mathrm{BLT}(\mathit{nil})\,.$$

However, if we consider the context $\mathit{IsEmpty}(\Box)$, then we have:

$$\mathrm{BLT}(\mathit{IsEmpty}(\bot)) = \{\bot\} \neq \{\bot, \mathit{True}\} = \mathrm{BLT}(\mathit{IsEmpty}(\mathit{nil}))\,.$$

Hence, we have defined a Böhm-like tree based on an ωTRS which does *not* satisfy congruence of Böhm-like tree equality.

Bibliographic Notes. As explained in Section 5.5, the Böhm-like trees defined by Boudol [Bou85], Blom [Blo01], and Ariola and Blom [AB02] generalise the Böhm-like trees defined in previous chapter. Hence, congruence does not hold for the trees defined by Boudol, Blom, and Ariola and Blom.

The above is contrary to the Böhm-like trees based on infinitary rewriting, as defined by Kennaway, Van Oostrom, and De Vries [KOV99]. These trees do satisfy congruence, as an immediate consequence of the fact that confluence holds for the rewrite systems employed to define the trees (see Chapter 7).

Remark that the above implies that not every Böhm-like tree defined along the lines of Chapter 5 can be defined as a Böhm-like tree in the sense of Kennaway, Van Oostrom, and De Vries. Their Böhm-like trees are always congruent, while this does not necessarily hold for the ones defined in Chapter 5.

6.2 Direct Approximant Functions

To obtain Böhm-like trees which are congruent, we strengthen the definition of direct approximant functions by adding the following to the definition, where s and t are terms and where $C[\Box]$ is a context:

If it holds that:

1. *for all* $s \to^* s'$ *there are* t' *with* $t \to^* t'$ *and* $\omega(s') \preccurlyeq \omega(t')$*, and*
2. *for all* $t \to^* t'$ *there are* s' *with* $s \to^* s'$ *and* $\omega(t') \preccurlyeq \omega(s')$*,*

then it also holds that:

1. *for all* $C[s] \to^* s'$ *there are* t' *with* $C[t] \to^* t'$ *and* $\omega(s') \preccurlyeq \omega(t')$*, and*
2. *for all* $C[t] \to^* t'$ *there are* s' *with* $C[s] \to^* s'$ *and* $\omega(t') \preccurlyeq \omega(s')$*.*

The above is simply the definition of congruence in disguised form: The definition of Böhm-like trees is substituted for BLT and the contents Proposition 3.2.10 is substituted for equality. Hence, congruence is immediately implied. As such, adding the above to the definition of direct approximant functions is only deceptively simple: The proof obligations stay the same.

Notwithstanding the deceptive simplicity, it is easy to show that the trivial trees from Example 5.2.9 and the normal form trees from Example 5.2.10 satisfy the above statement. Hence, both trees are congruent with respect to Böhm-like tree equality. Berarducci-like trees also satisfy the statement, since they satisfy congruence, as remarked in the previous section and as proved in Chapter 7.

Discussion. Obviously, a statement that allows for simpler proof obligations would be welcome in the case of direct approximant functions. Unfortunately, it is unclear to the author what such statements should look like.

To find statements that give to easier proof obligations, one might try to find overlap criteria such as those formulated in Section 6.4 with respect to ωTRSs and syntactic continuity. However, care must be taken in doing so. One might end up with Böhm-like trees that satisfy syntactic continuity. This is undesirable, since syntactic continuity is not satisfied by Berarducci-like trees, as we show in the next section.

6.3 Syntactic Continuity

To be able to ease the proof obligations regarding congruence, we next show that syntactic continuity implies both congruence and precongruence. In addition, we show that neither syntactic continuity nor precongruence necessarily holds for the Böhm-like trees defined in Chapter 5. More in particular, we show this for Berarducci-like trees.

The definition of precongruence is identical to the definition of congruence, except that the prefix order on infinite terms is substituted for equality. As such, it is easy to show that precongruence implies congruence. We discuss precongruence, because in the case of the Böhm and Lévy-Longo trees of the $\lambda\beta$-calculus it is often shown that syntactic continuity implies precongruence before it is shown that congruence is implied by precongruence.

6.3.1 Congruence

As claimed above, congruence is implied by *syntactic continuity*. That is, given a TRS $\mathcal{R} = (\Sigma, R)$ and a Böhm-like tree for $\mathcal{R}$, congruence is implied whenever it is assumed that:

$$\mathrm{BLT}(C[s]) = \bigsqcup\{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}\,,$$

where $s \in \mathcal{T}er(\Sigma_\perp, V)$ and where $C[\Box]$ is a context. We prove the claim:

Lemma 6.3.1. *Let $\mathcal{R} = (\Sigma, R)$ be a TRS and let* BLT *be a Böhm-like tree for $\mathcal{R}$. If the Böhm-like tree satisfies syntactic continuity, then Böhm-like tree equality with respect to* BLT *is a congruence.*

Proof. Assume syntactic continuity and let $s_1, s_2 \in \mathcal{T}er(\Sigma_\perp, V)$, $C[\Box]$ a context, and $\mathrm{BLT}(s_1) = \mathrm{BLT}(s_2)$. By syntactic continuity, the following hold:

$$\mathrm{BLT}(C[s_1]) = \bigsqcup\{\mathrm{BLT}(C[t_1]) \mid t_1 \in \mathrm{BLT}(s_1)\}$$
$$\mathrm{BLT}(C[s_2]) = \bigsqcup\{\mathrm{BLT}(C[t_2]) \mid t_2 \in \mathrm{BLT}(s_2)\}$$

Moreover, by the assumption that $\mathrm{BLT}(s_1) = \mathrm{BLT}(s_2)$, we have:

$$\bigsqcup\{\mathrm{BLT}(C[t_1]) \mid t_1 \in \mathrm{BLT}(s_1)\} = \bigsqcup\{\mathrm{BLT}(C[t_2]) \mid t_2 \in \mathrm{BLT}(s_2)\}\,.$$

Combining the two facts gives:

$$\begin{aligned}\mathrm{BLT}(C[s_1]) &= \bigsqcup\{\mathrm{BLT}(C[t_1]) \mid t_1 \in \mathrm{BLT}(s_1)\}\\ &= \bigsqcup\{\mathrm{BLT}(C[t_2]) \mid t_2 \in \mathrm{BLT}(s_2)\}\\ &= \mathrm{BLT}(C[s_2])\,.\end{aligned}$$

Hence, $\mathrm{BLT}(C[s_1]) = \mathrm{BLT}(C[s_2])$, as required. □

Obviously, since congruence generally does not hold (see Section 6.1), syntactic continuity generally does not hold either. To see this directly, without employing congruence, consider the Böhm-like tree we defined in Section 6.1. Given the term *nil* and the context $\mathit{IsEmpty}(\Box)$, we have:

$$\mathrm{BLT}(\mathit{IsEmpty}(\mathit{nil})) = \{\bot, \mathit{True}\}$$

and:

$$\bigsqcup\{\mathrm{BLT}(\mathit{IsEmpty}[t]) \mid t \in \mathrm{BLT}(\mathit{nil})\} = \bigsqcup\{\mathrm{BLT}(\mathit{IsEmpty}(\bot))\} = \{\bot\}\,.$$

Hence, since $\{\bot, \mathit{True}\} \neq \{\bot\}$, syntactic continuity does not hold.

Syntactic continuity does not necessarily hold for Berarducci-like trees either. Given the CL-term S and the context $\Box II(SII)$, we have:

$$\mathrm{BLT}(SII(SII)) = \{\bot\}\,.$$

and:

$$\begin{aligned}\bigsqcup\{\mathrm{BLT}(tII(SII)) \mid t \in \mathrm{BLT}(S)\} &= \bigsqcup\{\mathrm{BLT}(\bot II(SII)), \mathrm{BLT}(SII(SII))\} \\ &= \downarrow\{\bot II(SII)\}\,.\end{aligned}$$

Hence, since $\{\bot\} \neq \downarrow\{\bot II(SII)\}$, syntactic continuity does not hold for the Berarducci-like trees of CL. Recall in this respect that congruence *does* hold for Berarducci-like trees, as briefly mentioned in Section 6.1 (see also Chapter 7).

Bibliographic Notes. Since congruence does not hold for the Böhm-like trees defined by Boudol [Bou85], Blom [Blo01], and Ariola and Blom [AB02], as explained in Section 6.1, syntactic continuity holds neither. The same can be said about the Böhm-like trees defined by Kennaway, Van Oostrom, and De Vries [KOV99] by means of infinitary rewriting. In this case, syntactic continuity does not hold, because Berarducci-like trees can be defined (see Chapter 7).

6.3.2 Precongruence

We now discuss *precongruence* for Böhm-like trees. Precongruence is defined as follows, when s_1 and s_2 are terms and $C[\Box]$ is a context:

$$\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t) \text{ implies } \mathrm{BLT}(C[s]) \preccurlyeq \mathrm{BLT}(C[t])\,.$$

Obviously, precongruence implies congruence by definition of equality and the prefix order on infinite terms. Moreover, we can prove the following:

Proposition 6.3.2. *Let $\mathcal{R} = (\Sigma, R)$ be a TRS and let* BLT *be a Böhm-like tree for $\mathcal{R}$. If the Böhm-like tree satisfies syntactic continuity, then it also satisfies precongruence.*

Proof. The proof is similar to that of Lemma 6.3.1. Thus, let $s_1, s_2 \in \mathcal{T}er(\Sigma_\perp, V)$, $C[\Box]$ a context, and $\mathrm{BLT}(s_1) \preccurlyeq \mathrm{BLT}(s_2)$. By syntactic continuity, we have:

$$\mathrm{BLT}(C[s_1]) = \bigsqcup\{\mathrm{BLT}(C[t_1]) \mid t_1 \in \mathrm{BLT}(s_1)\}$$
$$\mathrm{BLT}(C[s_2]) = \bigsqcup\{\mathrm{BLT}(C[t_2]) \mid t_2 \in \mathrm{BLT}(s_2)\}$$

Moreover, by $\mathrm{BLT}(s_1) \preccurlyeq \mathrm{BLT}(s_2)$, it follows that:

$$\{\mathrm{BLT}(C[t_1]) \mid t_1 \in \mathrm{BLT}(s_1)\} \subseteq \{\mathrm{BLT}(C[t_2]) \mid t_2 \in \mathrm{BLT}(s_2)\}\,.$$

Hence, we have:

$$\bigsqcup\{\mathrm{BLT}(C[t_1]) \mid t_1 \in \mathrm{BLT}(s_1)\} \preccurlyeq \bigsqcup\{\mathrm{BLT}(C[t_2]) \mid t_2 \in \mathrm{BLT}(s_2)\}\,.$$

Combining the two facts gives:

$$\begin{aligned}\mathrm{BLT}(C[s_1]) &= \bigsqcup\{\mathrm{BLT}(C[t_1]) \mid t_1 \in \mathrm{BLT}(s_1)\}\\ &\preccurlyeq \bigsqcup\{\mathrm{BLT}(C[t_2]) \mid t_2 \in \mathrm{BLT}(s_2)\}\\ &= \mathrm{BLT}(C[s_2])\end{aligned}$$

Hence, $\mathrm{BLT}(C[s_1]) \preccurlyeq \mathrm{BLT}(C[s_2])$, as required. □

Although, syntactic continuity implies precongruence, just as it implies congruence, we do not want *every* Böhm-like tree, or even every denotational semantics, to satisfy precongruence. This, for example, would no longer allow us to define Berarducci-like trees. In other words, precongruence does not hold for Berarducci-like trees.

To see that precongruence does not hold for Berarducci-like trees, consider the CL-terms $\perp$ and SII and the context $\Box(SII)$. Obviously, we have:

$$\mathrm{BLT}_{\mathrm{BeL}}(\perp) = \downarrow\{\perp\} \preccurlyeq \downarrow\{SII\} = \mathrm{BLT}_{\mathrm{BeL}}(SII)\,.$$

However, we also have:

$$\mathrm{BLT}_{\mathrm{BeL}}(\perp(SII)) = \downarrow\{\perp(SII)\} \not\preccurlyeq \downarrow\{\perp\} = \mathrm{BLT}_{\mathrm{BeL}}(SII(SII))\,.$$

Remark that this also implies that precongruence does not follow from congruence.

Remark 6.3.3. A counterexample similar to the one given above exists in case of the Berarducci trees of the $\lambda\beta$-calculus[†]. In that case, the terms $\perp$ and $\lambda x.xx$ and the context $\Box(\lambda x.xx)$ suffice, since we have:

$$\mathrm{BLT}_{\mathrm{Be}}(\perp) = \downarrow\{\perp\} \preccurlyeq \downarrow\{\lambda x.xx\} = \mathrm{BLT}_{\mathrm{Be}}(\lambda x.xx)\,,$$

but also:

$$\mathrm{BLT}_{\mathrm{Be}}(\perp(\lambda x.xx)) = \downarrow\{\perp(\lambda x.xx)\} \not\preccurlyeq \downarrow\{\perp\} = \mathrm{BLT}_{\mathrm{Be}}((\lambda x.xx)(\lambda x.xx))\,.$$

[†]Previously, the author [Ket04, Section 8] was under the impression that precongruence *does* hold for Berarducci trees. The falsehood of this statement was pointed out to the author by De Vries [Vri04].

6.4 Direct Approximant TRSs

As explained in the previous section, syntactic continuity generally does not hold for Böhm-like trees defined by either a direct approximant function or an ωTRS. However, it could be argued that the presented counterexample involving an ωTRS is quite unreasonable: The Böhm-like tree of *nil* should not be $\{\bot\}$. Instead, it should be $\{\bot, nil\}$, as *nil* is a normal form and a nullary function symbol. In addition, the counterexample is also unreasonable since congruence is not satisfied.

The counterexample regarding Berarducci-like trees is more reasonable, since the Berarducci-like tree of any normal form n is $\downarrow\{n\}$. Moreover, Berarducci-like trees *do* satisfy congruence, as briefly mentioned in the previous section.

In the current section, we strengthen the definition of ωTRSs such that syntactic continuity of Böhm-like trees is implied. Remark that this does not affect the ability to define Berarducci-like trees: These trees can only de defined by means of a direct approximant function and not by means of an ωTRS.

The strengthened definition of ωTRSs and the intuition behind the strengthening occur in Section 6.4.1. In Section 6.4.2, we prove that the definition actually implies syntactic continuity. Finally, in Section 6.4.3, an open problem related to strengthening of the definition of ωTRSs is discussed.

Before we continue, remark that the counterexample for ωTRSs, as it occurs in the previous section, is in fact only a counterexample to:

$$\mathrm{BLT}(C[s]) \preccurlyeq \bigsqcup\{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}\,.$$

The reverse:

$$\mathrm{BLT}(C[s]) \succcurlyeq \bigsqcup\{\mathrm{BLT}(C[t]) \mid s \in \mathrm{BLT}(s)\}$$

always holds given a Böhm-like tree based on an ωTRS. This is the contents of the following proposition:

Proposition 6.4.1. *Let $\mathcal{R} = (\Sigma, R)$ be a confluent, left-linear TRS and let $\mathcal{D} = (\Sigma_\bot, D)$ be an ωTRS for $\mathcal{R}$. If $s \in \mathcal{T}er(\Sigma_\bot, V)$ and $C[\Box]$ a context, then the Böhm-like tree of $C[s]$ satisfies:*

$$\mathrm{BLT}(C[s]) \succcurlyeq \bigsqcup\{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}\,.$$

Proof. Let $s \in \mathcal{T}er(\Sigma_\bot, V)$ and $C[\Box]$ a context. If $t \in \mathrm{BLT}(s)$, then by the definition of Böhm-like trees there exists a term s' such that $s \to^* s'$ and $t \preccurlyeq \omega(s')$. Thus, as $\omega(s') \preccurlyeq s'$, we have $t \preccurlyeq s'$ and $C[t] \preccurlyeq C[s']$.

By monotonicity of Böhm-like trees based on ωTRSs (see Lemma 5.3.1 and Theorem 5.4.10), we have $\mathrm{BLT}(C[t]) \preccurlyeq \mathrm{BLT}(C[s'])$. Hence, by preservation of Böhm-like trees under rewriting, $\mathrm{BLT}(C[t]) \preccurlyeq \mathrm{BLT}(C[s']) = \mathrm{BLT}(C[s])$ and $\mathrm{BLT}(C[s])$ is an upper bound of the set:

$$\{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}\,.$$

By the above facts and the fact that $\mathrm{BLT}(s)$ is directed, it follows that the above set is directed. Hence, the set also has a least upper bound. The least upper bound must be smaller than or equal to $\mathrm{BLT}(C[s])$, which is an upper bound, as we just established. □

Bibliographic Notes. The strengthening of the definition of ωTRSs and the proof of syntactic continuity given below are based on Ariola's proof [Ari96] for Huet-Lévy trees. In turn, Ariola's proof is based on Lévy's proof [Lév75] for Lévy-Longo trees. Note that Barendregt's proof [Bar84] for Böhm trees is also based on Lévy's proof.

6.4.1 Strengthening

Syntactic continuity fails for the counterexample in Section 6.3 for one very particular reason: Both the considered terms have $\bot$ as their direct approximant, while placing the terms in a context yields a reduction to a term whose direct approximant is not $\bot$ in only one case. Avoiding this discrepancy allows us to prove syntactic continuity, as we explain below.

Given that we want to define an ωTRS for some TRS $\mathcal{R}$, there are at least two ways to avoid the above discrepancy:

- Modify $\mathcal{R}$ such that placing any two terms with $\bot$ as their direct approximant in a context yields either a redex in both cases or no redex at all.
- Make certain that the direct approximant of any reduct capable of causing the discrepancy is $\bot$, i.e., strengthen the definition of ωTRSs.

Modifying $\mathcal{R}$ is inappropriate: It makes it very difficult to the compare the different Böhm-like trees for $\mathcal{R}$, as modification may change, e.g., the set of root-stable terms. Hence, we opt for the second approach.

Observe that creating a redex by placing a term in a context while its direct approximant is equal to $\bot$ implies that there is *non-root overlap* between a rule of $\mathcal{R}$ and a rule of the ωTRS. For this reason, we strengthen the definition of ωTRSs by restricting overlap.

It is not necessary to rule out every form of non-root overlap. If the reduct of a term placed in a context $C[\Box]$, that is responsible for overlap, already has $\bot$ as direct approximant, then no restriction is necessary, because $\bot$ is also the direct approximant of $C[\bot]$ by the third and fourth clause of Definition 5.2.1. This form of overlap occurs not only in pathological examples: In the $\lambda\beta$-calculus, such overlap can be observed in case of the β-rule and $\lambda x.\bot \to_\omega \bot$, i.e., one of the rules of the rewrite system that defines the Böhm direct approximant. We have $\omega_{\mathrm{BT}}((\lambda x.\bot)s) = \omega_{\mathrm{BT}}(\bot s) = \bot$.

Remark 6.4.2. Besides the overlap described above, which is of the form $l[d]_p$ for some rule $l \to r$ from $\mathcal{R}$ and some rule $d \to_\omega e$ from the ωTRS, the definition given below also restricts overlap of the form $d[l]_p$. This second restriction is heavily employed in the proof of Lemma 6.4.14. However, unlike in the case of overlap of the form $l[d]_p$, the author did not succeed in constructing a counterexample that shows that overlap of the form $d[l]_p$ can yield a Böhm-like tree that is neither congruent nor syntactic continuous. Hence, the following conjecture:

Conjecture 6.4.3. Syntactic continuity, which holds for ωTRSs that satisfy Definitions 6.4.4 and 6.4.5, as shown in Theorem 6.4.18, also holds when the fourth clause of Definition 6.4.5 is omitted.

With respect to the strengthened definition of ωTRSs, as given below, we assume that $\mathcal{R} = (\Sigma, R)$ is an *orthogonal TRS*. This is contrary to the assumptions made in previous chapter, where we only required $\mathcal{R}$ to be a confluent, left-linear TRS. The change to orthogonal systems is further discussed in Section 6.4.3.

We strengthen the definition of ωTRSs by means of two TRSs. The first TRS, called the *redex removal* TRS, is used to replace the redexes of $\mathcal{R}$ by $\bot$. The second TRS, called the *melting* TRS, is essentially an ωTRS adapted in two ways: First, the clause that requires for each $l \to r \in R$ that l reduces to $\bot$ is removed, as it is dealt with by the redex removal TRS. Second, two clauses are added with respect to overlap between the rules of $\mathcal{R}$ and those of the ωTRS.

We start with the definition of the redex removal TRS:

Definition 6.4.4. *The* redex removal TRS (ω_{r}TRS) *for* $\mathcal{R}$ *is the TRS* $\mathcal{L} = (\Sigma_\bot, L)$ *with:*

$$L = \{l \to_{\mathrm{r}} \bot \mid l \to r \in R\}\,,$$

and whose rewrite relation is denoted $\to_{\mathrm{r}}$.

By orthogonality of $\mathcal{R}$, it follows immediately that $\mathcal{L}$ is orthogonal and, hence, confluent. Moreover, by the fact that each rewrite rule is of the form $l \to_\omega \bot$, we have that $\mathcal{L}$ is terminating. Whence, each term has a unique normal form with respect to the ω_{r}TRS.

The above definition defines a unique ω_{r}TRS for each orthogonal TRS $\mathcal{R}$. The freedom needed to define multiple Böhm-like trees is provided by the definition of melting TRSs:

Definition 6.4.5. *A* melting TRS (ω_{m}TRS) *for* $\mathcal{R}$ *is a left-linear TRS* $\mathcal{M} = (\Sigma_\bot, M)$*, whose rewrite relation, denoted* $\to_{\mathrm{m}}$*, satisfies the following conditions for all* $d \to_{\mathrm{m}} e \in M$ *and* $l \to r \in R$*:*

1. $e = \bot$,
2. $\bot$ *is a normal from with respect to* $\to_{\mathrm{m}}$,
3. $s \to_{\mathrm{m}}^* \bot$ *for all* s *such that*
   - $s \preccurlyeq d$, *or*
   - $s \preccurlyeq l$ *with* s *not an* $l \to r$*-redex,*
4. $\sigma(d[r]_p) \to_{\mathrm{m}}^* \bot$*, if* l *overlaps* d *at* $p \in \mathcal{P}os(d)$ *with* σ *as mgu (see Figure 6.1),*
5. $\sigma(r) \to_{\mathrm{m}}^* \bot$*, if* d *overlaps* l *at* $p \in \mathcal{P}os(l)$ *with* σ *as mgu (see Figure 6.2).*

We call the TRSs defined above melting TRSs, as their transitive-reflexive closures contain what are sometimes called the *melting rules*: all the rules of the form $s \to \bot$ where $s \preccurlyeq l$ with $l \to r \in R$ and s not an $l \to r$-redex. The presence of the melting rules is immediate by the third clause of the definition.

Remark 6.4.6. The requirement $s \preccurlyeq l$ with s not an $l \to r$-redex in the third clause of Definition 6.4.5 ensures that s is not a substitution instance of l. We do not require substitution instances to be included in ω_{m}TRSs, as these are dealt with by ω_{r}TRSs. Of course, the definition does not forbid the inclusion substitution instances. However, inclusion creates obligations with respect to the fifth clause of the definition.

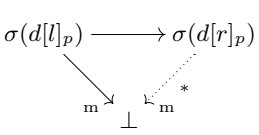

Figure 6.1. Definition 6.4.5.(4)

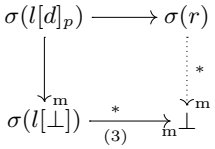

Figure 6.2. Definition 6.4.5.(5)

The requirement also implies that s does *not* overlap with *any* rewrite rule of $\mathcal{R}$. If overlap occurs with the left-hand side l' of a rule of $\mathcal{R}$, then there is also overlap between l and l' by the fact that $s \preccurlyeq l$. This is impossible by orthogonality of $\mathcal{R}$.

The following holds with respect to ω_{r}TRSs and ω_{m}TRSs:

Proposition 6.4.7. *Let $\mathcal{L} = (\Sigma_\bot, L)$ be the ω_{r}TRS for $\mathcal{R}$. If $\mathcal{M} = (\Sigma_\bot, M)$ is an ω_{m}TRS for $\mathcal{R}$, then $\mathcal{LM} = (\Sigma_\bot, L \cup M)$ is an ωTRS for $\mathcal{R}$.*

Proof. This is immediate by the definition of ω_{r}TRSs and the first three clauses of Definition 6.4.5. □

Hence, the combination of ω_{r}TRSs and ω_{m}TRSs strengthens the definition of ωTRSs.

Example 6.4.8 (Huet-Lévy ω_{m}TRS). For an orthogonal TRS $\mathcal{R} = (\Sigma, R)$, the Huet-Lévy ωTRS can be redefined as a combination of an ω_{r}TRS and an ω_{m}TRS. The rewrite rules of the ω_{m}TRS are the rules $s \to_{\mathrm{m}} \bot$ such that $s \preccurlyeq l$ with $l \to r \in R$ and s not an $l \to r$-redex. The rules are readily shown to satisfy all clauses of Definition 6.4.5, due to the lack of overlap with $\mathcal{R}$. That the transitive-reflexive of the union of the ω_{r}TRS and the ω_{m}TRS is identical to the transitive-reflexive closure of the Huet-Lévy ωTRS follows easily.

It must be remarked that the Huet-Lévy ωTRS defined by Ariola [Ari96] actually consists of an ω_{r}TRS and an ω_{m}TRS and not only of an ωTRS as suggested in the previous chapter.

Example 6.4.9. The ωTRS for CL, as presented in the discussion at the end of Section 5.4 can also be redefined as a combination of an ω_{r}TRS and an ω_{m}TRS. The ω_{r}TRS consists of the following rewrite rules:

$$\begin{aligned} Sxyz &\to_{\mathrm{r}} \bot \\ Kxy &\to_{\mathrm{r}} \bot \\ Ix &\to_{\mathrm{r}} \bot \end{aligned}$$

and the ω_{m}TRS has the rules:

$$\begin{aligned} \bot x &\to_{\mathrm{m}} \bot \\ S\bot &\to_{\mathrm{m}} \bot \\ K\bot &\to_{\mathrm{m}} \bot \end{aligned}$$

The first three clauses of Definition 6.4.5 are easily verified for the above ω_{m}TRS. The same holds for the fourth clause, as no overlap of its kind occurs. That the fifth clause also holds follows from the reductions depicted in Figure 6.3.

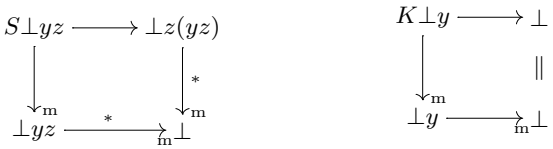

Figure 6.3. Example 6.4.9

Remark 6.4.10. Although the $\lambda\beta$-calculus is not a TRS, we can also redefine the direct approximant rewrite systems of the Böhm and Lévy-Longo trees as combinations of a redex removal rewrite system and a melting rewrite system. In both cases, the redex removal rewrite system consists of the rule:

$$(\lambda x.s)t \to_{\mathrm{r}} \bot\,,$$

and the melting rewrite system contains the rule:

$$\bot t \to_{\mathrm{m}} \bot\,.$$

In case of Böhm trees, the melting rewrite system also contains the rule:

$$\lambda x.\bot \to_{\mathrm{m}} \bot\,.$$

That each of the melting rewrite systems satisfies all clauses of Definition 6.4.5 is easily verified.

By virtue of the first three clauses of Definition 6.4.5, we have that most properties of ωTRSs carry over to ω_{m}TRSs. The exceptions are Lemmas 5.4.8 and 5.4.9.(5) and Theorem 5.4.10. They do not carry over, as they depend on the fourth clause of Definition 5.4.1, which does not occur in Definition 6.4.5.

Assuming that $\mathcal{M} = (\Sigma_\bot, M)$ is an arbitrary ω_{m}TRS, we next summarise the properties from Section 5.4 that carry over to ω_{m}TRSs.

Lemma 6.4.11. *The ω_{m}TRS $\mathcal{M}$ is confluent and normalising.*

Proof. Identical to the proofs of Lemmas 5.4.4 and 5.4.6. □

By the previous lemma, we have that each term s has a unique normal form with respect to the ω_{m}TRS $\mathcal{M}$. We denote the unique normal form by $\omega_{\mathrm{m}}(s)$. By the fact that the lemma does not depend on the fourth and fifth clause of Definition 6.4.5, it is decidable whether the fourth and fifth clause are satisfied. Hence, it is also decidable whether a TRS is an ω_{m}TRS.

Continuing with the properties from Section 5.4, we have:

Lemma 6.4.12. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\bot, V)$. If $s \preccurlyeq t$ and $t \to_{\mathrm{m}}^* t'$, then there exist $s' \in \mathcal{T}er(\Sigma_\bot, V)$ such that $s' \preccurlyeq t'$ and $s \to_{\mathrm{m}}^* s'$.*

Proof. Identical to the proof of Lemma 5.4.7. □

Finally, we have:

Lemma 6.4.13. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$. The following properties hold:*

1. $\omega_{\mathrm{m}}(s) \preccurlyeq s$,
2. $\omega_{\mathrm{m}}(s) = \omega_{\mathrm{m}}(s[\omega_{\mathrm{m}}(s|_p)]_p)$ *for all* $p \in \mathcal{P}os(s)$,
3. $\omega_{\mathrm{m}}(\omega(s)) = \omega_{\mathrm{m}}(s)$, *and*
4. $\omega_{\mathrm{m}}(s) \preccurlyeq \omega_{\mathrm{m}}(t)$, *if* $s \preccurlyeq t$.

Proof. Identical to the proofs of the first four clauses of Lemma 5.4.9, where the appropriate analogues of the employed lemmas are substituted. □

Discussion. Separating ωTRSs in ω_{r}TRSs and ω_{m}TRSs emphasizes the fact that not all terms that have $\perp$ as their direct approximant can be further reduced. This is the case when the rules of the ω_{r}TRS do not apply. The separation also facilitates the proof of syntactic continuity. In particular, it is employed in the proof of Lemma 6.4.17.

Separation in ω_{r}TRSs and ω_{m}TRSs also has a disadvantage: ω_{r}TRSs are only confluent because we assume that $\mathcal{R}$ is orthogonal. Confluence does not hold for ω_{r}TRSs of arbitrary (confluent) left-linear TRSs, as explained in Section 6.4.3.

6.4.2 Syntactic Continuity

In this section, we prove that ωTRSs defined by means of ω_{r}TRSs and ω_{m}TRSs yield Böhm-like trees that satisfy syntactic continuity. Essentially, we show that $s \to_{\mathrm{m}}^* t$ implies $\mathrm{BLT}(s) = \mathrm{BLT}(t)$, from which syntactic continuity follows with relative ease.

As in the previous section, we assume that $\mathcal{R} = (\Sigma, \perp)$ is an orthogonal TRS. In addition, we assume that $\mathcal{L} = (\Sigma_\perp, L)$ is the ω_{r}TRS for $\mathcal{R}$ and that $\mathcal{M} = (\Sigma_\perp, M)$ is an ω_{m}TRS for $\mathcal{R}$. Recall that the union $\mathcal{LM} = (\Sigma_\perp, L \cup M)$ of $\mathcal{L}$ and $\mathcal{M}$ is an ωTRS. As usual, we denote the unique normal form of a term s with respect to the ωTRS by $\omega(s)$.

To show that $s \to_{\mathrm{m}}^* t$ implies $\mathrm{BLT}(s) = \mathrm{BLT}(t)$, we first prove two lemmas relating the rewrite steps of $\mathcal{R}$ and the ω_{m}TRS.

Lemma 6.4.14. *Let $s, s', t \in \mathcal{T}er(\Sigma_\perp, V)$. If $s' \leftarrow s \to_{\mathrm{m}}^* t$, then there exist $t'_1, t'_2 \in \mathcal{T}er(\Sigma_\perp, V)$ such that $s' \to_{\mathrm{m}}^* t'_2 \,{}^{=}\!\leftarrow t'_1 \,{}^{*}_{\mathrm{m}}\!\leftarrow t$ (see Figure 6.4).*

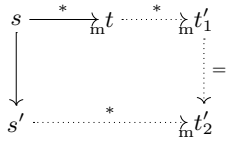

Figure 6.4. Lemma 6.4.14

Proof. Suppose $s' \leftarrow s \to^*_{\mathrm{m}} t$ such that p is the position of the redex contracted in $s \to s'$ and such that $l \to r \in R$ is the employed rewrite rule. There are two cases to consider depending on the occurrence of a redex at position p in t.

A redex occurs at position p in t. By left-linearity of $\mathcal{R}$ and the first clause of Definition 6.4.5, we have $t|_p = \sigma(l)$ for some substitution σ. Define $t'_1 = t$ and $t'_2 = t[\sigma(r)]_p$. As $t|_p = \sigma(l) \to \sigma(r)$, we have $t'_1 \to t'_2$. Moreover, by the fact that a redex occurs at position p in t, we have for each redex contracted in $s \to^*_{\mathrm{m}} t$, where q is the position of the redex, that q does not occur in the redex pattern of the redex at position p. That is, either $p \parallel q$ or $q \geqslant p \cdot p'$ with $p' \in \mathcal{P}os(l)$ and $l|_{p'} \in \mathcal{V}ar(l)$. Hence, $s' \to^*_{\mathrm{m}} t'_2$.

No redex occurs at position p in t. Considering $s \to^*_{\mathrm{m}} t$ from left to right, there is either no step whose redex pattern overlaps with the redex pattern of the redex contracted in $s \to s'$ or there is a first step.

If there is no step, then there is obviously a position $q \leqslant p$ such that $t|_q = \bot$. Define $t'_1 = t'_2 = t$. By non-overlap and left-linearity of ω_{m}TRSs we have $s' \to^*_{\mathrm{m}} t'_2$. Moreover, $t'_1 \to^= t'_2$ is immediate by $t'_1 = t'_2$.

If there is a step with overlap, then there are again two possibilities. Assuming that $d \to_{\mathrm{m}} e \in M$ is the rewrite rule employed in the step, either l overlaps d or d overlaps l. In the first case, the result follows by defining $t'_1 = t$ and $t'_2 = t$ and by employing the third and fourth clause of Definition 6.4.5 in the construction of $s' \to^*_{\mathrm{m}} t'_2$.

In the second case, note that other redexes may be contracted that overlap with the prefix of redex pattern of l as it exists after the step contracting the first overlapping redex. This does not pose a problem by the third clause of Definition 6.4.5. If $p \in \mathcal{P}os(t)$ the result follows by defining $t'_1 = t[\bot]_p$ and $t'_2 = t[\bot]_p$ and by employing the third clause of Definition 6.4.5 in the construction of $t \to^*_{\mathrm{m}} t'_1$ and the fifth clause in the construction of $s' \to^*_{\mathrm{m}} t'_2$. Otherwise, if $p \notin \mathcal{P}os(t)$ the result follows by defining $t'_1 = t'_2 = t$ and by employing third and fifth clause of Definition 6.4.5 in the construction of $s' \to^*_{\mathrm{m}} t'_2$. □

Lemma 6.4.15. *Let $s, s', t \in \mathcal{T}er(\Sigma_\bot, V)$. If $s' \,{}^*\!\leftarrow s \to^*_{\mathrm{m}} t$, then there exist $t' \in \mathcal{T}er(\Sigma_\bot, V)$ such that $t \to^* t'$ and $\omega(s') = \omega(t')$ (see Figure 6.5).*

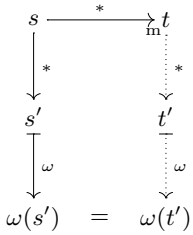

Figure 6.5. Lemma 6.4.15

Proof. Suppose $s' \mathrel{{}^*\!\leftarrow} s \to^*_{\mathrm{m}} t$. By Lemma 6.4.14 we can erect Figure 6.6. Moreover, by left-linearity of $\mathcal{R}$ and the definition of ω_{m}TRSs there exists a term t''_2 such that $t \to^= t''_2$, contracting a redex which has the redex contracted in $t'_1 \to^= t'_2$ as residual. In case $t'_1 \to^= t'_2$ is empty we have $t'_1 = t'_2$ and we can define $t''_2 = t$ from which we obtain $t''_2 \to^*_{\mathrm{m}} t'_2$. In case $t'_1 \to^= t'_2$ is not empty, we also have that $t''_2 \to^*_{\mathrm{m}} t'_2$, by definition of ω_{m}TRSs and the fact that all reductions in $t \to^*_{\mathrm{m}} t'_2$ occur at positions outside the redex pattern of the redex contracted in $t \to^= t''_2$.

We can repeat the above construction of t''_2 for the terms t''_2, t'_3, and t'_4, and so on downwards up to t'_n. This yields a term $t' = t''_n$ such that $t \to^* t'$, as required. Moreover, as $t' \to^*_{\mathrm{m}} t'_n$ and $s' \to^*_{\mathrm{m}} t'_n$, it follows by definition of ωTRSs that $\omega(s') = \omega(t'_n) = \omega(t')$, as also required. □

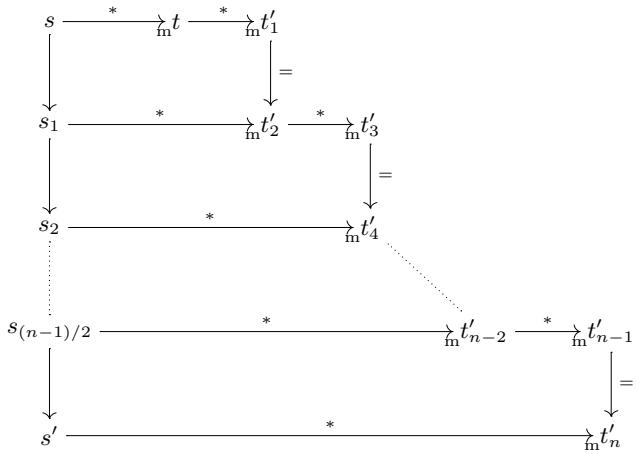

Figure 6.6. Proof of Lemma 6.4.15

We can now establish the desired relation between the ω_{m}TRS $\mathcal{M}$ and the Böhm-like tree BLT based on the ω_{m}TRS:

Lemma 6.4.16. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$. If $s \to^*_{\mathrm{m}} t$, then* $\mathrm{BLT}(s) = \mathrm{BLT}(t)$.

Proof. Suppose $s \to^*_{\mathrm{m}} t$. By Lemma 6.4.15 we have for all $s \to^* s'$ that there exist t' such that $t \to^* t'$ and $\omega(s') = \omega(t')$. Hence, $\mathcal{A}(s) \subseteq \mathcal{A}(t)$ and $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$. Moreover, as $s \succcurlyeq t$, we have by monotonicity of Böhm-like trees based on ωTRSs that $\mathrm{BLT}(s) \succcurlyeq \mathrm{BLT}(t)$ and we can conclude that $\mathrm{BLT}(s) = \mathrm{BLT}(t)$. □

We next prove that syntactic continuity holds for Böhm-like trees based on ω_{r}TRSs and ω_{m}TRSs. Of course, Proposition 6.4.1 already establishes half of the proof, which leaves the other half:

Lemma 6.4.17. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$ and let $C[\Box]$ be a context. The Böhm-like tree of $C[s]$ based on $\mathcal{LM}$ satisfies:*

$$\mathrm{BLT}(C[s]) \preccurlyeq \bigsqcup\{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}\,.$$

Proof. That the least upper bound of $\{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}$ exists is already shown in Proposition 6.4.1. Consequently, we only need to prove that there exists for every $C[s] \to^* s'$ a term $t \in \mathrm{BLT}(s)$ such that $\omega(s') \in \mathrm{BLT}(C[t])$. Thus, let $C[s] \to^* s'$. We erect Figure 6.7.

Square (1) exists by Lemma 2.2.9. From the lemma it follows immediately that there are terms t' and t'' such that $C[s] \to^*_{\mathrm{io}} C[t'] \to^*_{\mathrm{io}} t''$ and $s' \to^* t''$ and such that no redex contracted in $C[t'] \to^* t''$ occurs at a position which is a descendant of a position $p \geqslant q$ with q the position of the hole in $C[\Box]$.

In Square (2), the term $t'_\perp$ denotes the normal form of t' with respect to the ω_{r}TRS $\mathcal{L}$. By orthogonality of $\mathcal{R}$ and the fact that no redex contracted in $C[t'] \to^* t''$ occurs at a position $p \geqslant q$ with q the position of the hole in $C[\Box]$, it follows that $C[t'_\perp] \to^* t''_\perp \,{}^*_{\mathrm{r}}\!\leftarrow t''$. Hence, Square (2) exists.

Square (3) exist by the fact that the ωTRS $\mathcal{LM}$ defines a direct approximant function. Finally, Square (4) exist by the fact that $t'' \to^*_{\mathrm{r}} t''_\perp$ implies $t'' \to^*_\omega t''_\perp$ and the fact that ωTRSs are confluent.

Since $\omega(s') \preccurlyeq \omega(t''_\perp)$ and $C[t'_\perp] \to^* t''_\perp$, it follows that $\omega(s') \in \mathrm{BLT}(C[t'_\perp])$. Moreover, as there are by definition no redexes of $\mathcal{R}$ in $t'_\perp$, we have $t'_\perp \to^*_{\mathrm{m}} \omega(t'_\perp)$ and, by Lemma 6.4.16, $\mathrm{BLT}(C[t'_\perp]) = \mathrm{BLT}(C[\omega(t'_\perp)])$. Hence, $\omega(s') \in \mathrm{BLT}(C[\omega(t'_\perp)]) = \mathrm{BLT}(C[\omega(t')])$. As $s \to^* t'$, we have $\omega(t') \in \mathrm{BLT}(s)$ and the result follows by choosing $t = \omega(t')$. □

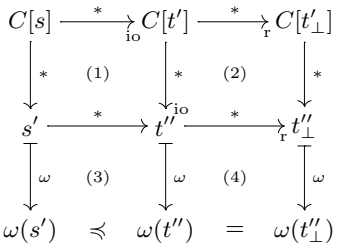

Figure 6.7. Proof of Lemma 6.4.17

The main theorem of this chapter is now immediate:

Theorem 6.4.18. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$ and let $C[\Box]$ be a context. The Böhm-like tree of $C[s]$ based on $\mathcal{LM}$ satisfies:*

$$\mathrm{BLT}(C[s]) = \bigsqcup\{\mathrm{BLT}(C[t]) \mid t \in \mathrm{BLT}(s)\}\,.$$

Proof. By Proposition 6.4.1 and Lemma 6.4.17. □

By the previous theorem and the redefinition of the Huet-Lévy ωTRS in Example 6.4.8, it follows that syntactic continuity is satisfied by Huet-Lévy trees. By Example 6.4.9, the same holds for the Böhm-like tree for CL as presented in the

discussion at the end of Section 5.4. In addition, it follows by Lemma 6.3.1 that both Huet-Lévy trees and the Böhm-like tree for CL are congruent with respect to Böhm-like tree equality.

6.4.3 Open Problem

With respect to congruence and ωTRSs, at least one question remains: Can we extend the strengthening of ωTRSs, as presented above, to arbitrary confluent, left-linear TRSs? We discuss this question.

At least three problems arise when extending the strengthening to arbitrary confluent, left-linear TRSs. The problems are related, respectively, to ω_{r}TRSs, inside-out reductions, and ω_{m}TRSs. We discuss each of the problems in turn.

Redex Removal TRSs. Given an arbitrary confluent, left-linear TRS, it does not always hold that its ω_{r}TRS is confluent. To see this, consider the left-linear TRS with the following two reduction rules:

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

The TRS is confluent, as it is weakly orthogonal: $\langle f(h(x)), f(h(x))\rangle$ is the only critical pair and this pair is trivial. The ω_{r}TRS for the TRS has the following rules:

$$f(g(x)) \to_{\mathrm{r}} \bot$$
$$g(h(x)) \to_{\mathrm{r}} \bot$$

The rules form non-confluent TRS, as is witnessed by the term $f(g(h(x)))$: The reducts of the term are $f(\bot)$ and $\bot$, which are normal forms.

There is a very easy solution to overcome the non-confluence of ω_{r}TRSs: Introduce a redex removal *function*. Inspired by the second clause of the definition of direct approximant functions, we could define the redex removal function for each term $s \in \mathcal{T}er(\Sigma_\bot, V)$ as the largest term t with respect to the prefix order such that $t \preccurlyeq s[\bot]_p$ for all $p \in \mathcal{P}os(s)$ with p the position of a redex in s.

In case of an orthogonal TRS, the value assigned to a term by a redex removal function is obviously identical to the unique normal form with respect to the ω_{r}TRS. In case of an arbitrary confluent, left-linear TRS, the value corresponds to the replacement of all outermost redexes by $\bot$.

Inside-Out Reductions. A second problem with extending the strengthening of ωTRSs to arbitrary confluent, left-linear TRSs is related to inside-out reductions, as employed in the proof of Lemma 6.4.17. In general, inside-out reductions do not exist in the case of arbitrary confluent, left-linear TRSs, as pointed out to the author by Van Oostrom [Oos04]. To see this, consider the left-linear TRS with the following rewrite rules:

$$\begin{aligned} a &\to b \\ b &\to a \\ f(x) &\to g(x,x) \\ g(a,x) &\to h(x,x) \\ h(b,x) &\to c \end{aligned}$$

Moreover, consider the following reduction:

$$f(\underline{a}) \to g(\underline{a},\underline{a}) \to h(\underline{a},\underline{a}) \to h(b,\underline{a}) \to c\,,$$

where each underlining denotes a residual of the redex a that occurs in $f(a)$. In the first two steps of the reduction, residuals of the redex a occur inside the contracted redexes. Hence, as a residual of the redex a is eventually contracted, the reduction is not inside-out.

In the vein of Lemma 2.2.9, we would now like to obtain an inside-out reduction from $f(a)$ to some term s such that $c \to^* s$. In fact, since c is a normal form of the considered TRS, we would actually like to obtain an inside-out reduction from $f(a)$ to c. We next try to construct such an inside-out reduction.

By definition of the considered TRS, and especially the rewrite rule $h(b,x) \to c$, the inside-out reduction must be of the form:

$$f(a) \to^* h(b,s) \to c\,,$$

for some term s. The subterm b of $h(b,s)$ must have been created in the reduction $f(a) \to^* h(b,s)$. By inspection of the TRS, we can see that the creating b is only possible by contraction of some residual of the redex a in $f(a)$. Hence, as we are trying to construct an inside-out reduction, the first step should be $f(a) \to f(b)$.

Having reduced $f(a)$ to $f(b)$ there are next two possibilities: Either reduce $f(b)$ to $f(a)$ or reduce it to $g(b,b)$. As the first possibility brings us back our original term, the only real possibility is to reduce $f(b)$ to $g(b,b)$. Next, $g(b,b)$ must either be reduced to $g(a,b)$ or to $g(b,a)$. Choosing $g(b,a)$ only allows for a reduction back to $g(b,b)$ or contraction of the redex b, which is a residual of one of redexes in $g(b,b)$. Hence, again there is only one real possibility: Contract the leftmost b redex in $g(b,b)$. Doing so we obtain the term $g(a,b)$, which has an inside-out reduction to c:

$$g(a,b) \to h(b,b) \to c\,.$$

Summarising the above, we obtain the following reduction:

$$f(a) \to f(b) \to g(b,b) \to g(a,b) \to h(b,b) \to c\,.$$

The last two steps of the reduction are inside-out, while first three seem to be the only steps possible, given that we want to construct an inside-out reduction. Unfortunately, the reduction is not an inside-out reduction. The redex b contracted in the third step is a residual of the redex b in $f(b)$ and, relative to the redex contracted in the third step, we have that b is inside. Hence, the third step should be $f(b) \to f(a)$, which can only be followed by $f(a) \to f(b)$, etc. Of course, we now

have a cyclic reduction which never reaches the term c. Hence, we can conclude that no inside-out reduction exists in this case.

Remark that we have not yet shown that the above TRS is confluent. However, this follows easily from the correspondence with the following *orthogonal* TRS, which combines the nullary function symbols a and b into a nullary function symbol ab and which removes the first two rewrite rules:

$$\begin{aligned} f(x) &\to g(x,x) \\ g(ab,x) &\to h(x,x) \\ h(ab,x) &\to c \end{aligned}$$

Hence, we can conclude that left-linearity and confluence are not sufficient to guarantee the existence of inside-out reductions in the vein of Lemma 2.2.9. Of course, it may be possible to prove syntactic continuity without employing inside-out reductions.

Melting TRSs. A third problem that arises when trying to extend the strengthening of ωTRSs to arbitrary confluent, left-linear TRSs has to do with the definition of ω_{m}TRSs. Copying the definition verbatim does not guarantee congruence, even if we have a redex removal function, as proposed above, and even if we could circumvent the problems with inside-out reductions. To see this, consider again the confluent, left-linear TRS we employed to explain the non-existence of inside-out reductions. The following two rewrite rules form an ω_{m}TRS for the TRS, which is easily verified, as the last two clauses of Definition 6.4.5 do not apply.

$$\begin{aligned} g(\bot,x) &\to_{\mathrm{m}} \bot \\ h(\bot,x) &\to_{\mathrm{m}} \bot \end{aligned}$$

Consider the nullary function symbols $\bot$ and a and the context $f(\Box)$. Obviously, we have:

$$\begin{aligned} \mathrm{BLT}(\bot) &= \{\bot\} \\ \mathrm{BLT}(a) &= \{\bot\} \end{aligned}$$

Hence, $\mathrm{BLT}(\bot) = \mathrm{BLT}(a)$. However, by the reductions employed in the discussion of inside-out reductions, we have:

$$\mathrm{BLT}(f(a)) = \{\bot, c\}$$

and, by the reduction $f(\bot) \to g(\bot,\bot)$, where the second term is a normal form, we have:

$$\mathrm{BLT}(f(\bot)) = \{\bot\}\,.$$

Thus, $\mathrm{BLT}(f(a)) \neq \mathrm{BLT}(f(\bot))$ and we conclude for arbitrary confluent, left-linear TRSs that congruence does not hold for Böhm-like trees whose direct approximant function is based on a redex removal function and an ω_{m}TRS.

7

Infinitary Rewriting

Infinity itself looks flat and uninteresting. Looking up into the night sky is looking into infinity – distance is incomprehensible and therefore meaningless.

— Douglas Adams
The Hitchhiker's Guide to the Galaxy (1979)

In this chapter, we compare the Böhm-like trees for TRSs, as defined in the previous two chapters, to the Böhm-like trees defined by Kennaway, Van Oostrom, and De Vries [KOV99] via infinitary rewriting.

We show that each Böhm-like tree defined by means of infinitary rewriting can also be defined by means of a direct approximant function. We also show that the reverse does not hold in general, but that a number of specific Böhm-like trees defined by means of a direct approximant function *can* be defined by means of infinitary rewriting.

By the above results, it follows that Böhm-like trees based on direct approximant functions are more general than those defined by means of infinitary rewriting. Moreover, the results regarding the specific Böhm-like trees provide us with an elegant way of establishing congruence of Böhm-like tree equality, since Böhm-like trees defined by means of infinitary rewriting are always congruent.

This chapter is organised as follows: In Sections 7.1 and 7.2, we introduce, respectively, infinitary term rewriting and Böhm-like trees based infinitary term rewriting. Thereafter, in Section 7.3 we prove that each Böhm-like tree based on infinitary rewriting can also be defined by means of a direct approximant function. In Section 7.4, we show that the reverse of the statement proven in Section 7.3 does not hold. However, we do show that Böhm-like trees based on infinitary rewriting can be given in at least three particular instances. Finally, in Section 7.5, we summarise the results presented in this chapter together with some of the results presented in the previous two chapters.

Throughout this chapter we assume that TRSs are *orthogonal*, not just confluent and left-linear. The reason is twofold: First, Kennaway, Van Oostrom, and De Vries [KOV99, Section 7] assume orthogonality to obtain their desired confluence results. Second, orthogonality is also assumed in Chapter 6.

7.1 Infinitary Term Rewriting

We give a short overview of the notions and notation from infinitary term rewriting as required in this chapter. For more complete overviews the book by Terese [Ter03] and the paper by Kennaway, Klop, Sleep, and De Vries [KKSV95] may be consulted.

Before we can define infinitary term rewriting, it is first necessary to define infinite terms. Given a signature Σ and a countably infinite set of variables V, it usual practise in infinitary term rewriting to define the set of infinite terms as $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma, V)$. That is, as the metric completion of the set $\mathcal{T}er(\Sigma, V)$, where the employed metric is the term metric (see Section 3.4).

Given a fresh nullary function symbol $\bot$, it follows by the theory developed in Chapter 3 that $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma_{\bot}, V)$ is isomorphic to $\mathcal{T}er^{\infty}_{\mathrm{i}}(\Sigma_{\bot}, V)$, the set of infinite terms defined by means of ideal completion (see Section 3.2). Therefore, we permit ourselves in the remainder of this chapter to confuse the sets $\mathcal{T}er^{\infty}_{\mathrm{m}}(\Sigma_{\bot}, V)$ and $\mathcal{T}er^{\infty}_{\mathrm{i}}(\Sigma_{\bot}, V)$ and to denote them both by $\mathcal{T}er^{\infty}(\Sigma_{\bot}, V)$. In addition, we also permit ourselves to confuse the set $\mathcal{T}er(\Sigma_{\bot}, V)$ and the set:

$$\{\iota(s) \mid s \in \mathcal{T}er(\Sigma_{\bot}, V)\} \subseteq \mathcal{T}er^{\infty}(\Sigma_{\bot}, V)\,, \tag{7.1}$$

with ι the embedding of the terms into the infinite terms (see Section 3.2).

Substitutions for infinite terms are easily defined by coalgebraically extending the definition of substitutions for finite terms (see Chapter 2). Moreover, (infinite) contexts are easily defined by adding a fresh nullary function symbol $\Box$ to the signature.

We can now define infinitary rewrite rules and infinitary TRSs:

Definition 7.1.1. *An* infinitary rewrite rule *is a pair (l, r) with $l \in \mathcal{T}er(\Sigma, V)$ and $r \in \mathcal{T}er^{\infty}(\Sigma, V)$, denoted $l \to r$, such that $\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$ and l not a variable.*

An infinitary TRS (iTRS) *over a signature Σ is a pair $\mathcal{R} = (\Sigma, R)$ with R a set of infinitary rewrite rules.*

Remark that the left-hand side of an infinitary rewrite rule is a (finite) term. For this reason, the definitions of *left-linearity* and *orthogonality* carry over directly from TRSs. Moreover, note that each TRS is also an iTRS by the fact that $\mathcal{T}er(\Sigma, V)$ is isomorphic to the subset of $\mathcal{T}er^{\infty}(\Sigma, V)$ depicted in (7.1).

We can now define what it means for an infinite term S to be rewritten to an infinite term T:

Definition 7.1.2. *Let $l \to r$ be an infinitary rewrite rule. Given a substitution σ, the term $\sigma(l)$ is called an $l \to r$-*redex*. If $S = C[\sigma(l)]$ for some $l \to r$-redex and context $C[\Box]$ with $C[\Box]|_p = \Box$, then an $l \to r$-redex, or simply a redex,* occurs *at position p and depth $|p|$ in S. Moreover, if $q \in \mathcal{P}os(S)$ then q is said to occur in the* redex pattern *of the $l \to r$ redex at position p in S, whenever $q \geqslant p$ and not $q \geqslant p \cdot p'$ with $p' \in \mathcal{P}os(l)$ such that $l|_{p'} \in \mathcal{V}ar(l)$.*

A pair $(S, T) \in \mathcal{T}er^{\infty}(\Sigma, V) \times \mathcal{T}er^{\infty}(\Sigma, V)$, denoted $S \to T$, defines a rewrite step*, if $S = C[\sigma(l)]$, $T = C[\sigma(r)]$, and if $l \to r$ is an infinitary rewrite rule. An $l \to r$-redex is* contracted *in such a step.*

We next define transfinite reductions:

Definition 7.1.3. *A* transfinite reduction *of ordinal length α is a sequence of infinite terms $(S_\kappa)_{\kappa<\alpha+1}$ such that $S_\kappa \to S_{\kappa+1}$ for all $\kappa < \alpha$. For each rewrite step $S_\kappa \to S_{\kappa+1}$, let d_κ denote the depth of the contracted redex. The reduction is called*

weakly convergent *or* Cauchy convergent *if it is continuous in the sense of Definition 2.1.2. Furthermore, it is called* strongly convergent *if it is weakly convergent and if d_κ tends to infinity as κ approaches γ from below.*

Note that every weakly convergent reduction is strongly convergent. Moreover, note that all finite reductions, as defined in Chapter 2, are weakly convergent; the same cannot be said of all infinite reductions, as not every infinite reduction needs to have a limit, which is required here.

By $S \twoheadrightarrow^\alpha T$, respectively $S \twoheadrightarrow^{\leqslant\alpha} T$, we denote a *strongly convergent* transfinite reduction of ordinal length α, respectively of ordinal length less than or equal to α. By $S \twoheadrightarrow T$ we denote a *strongly convergent* transfinite reduction of arbitrary ordinal length.

An iTRS is *infinitary normalising*, if for each infinite term there exists a strongly convergent reduction sequence to a normal form, i.e., an infinite term in which no redex occurs. Moreover, an iTRS is *confluent* whenever $T \twoheadleftarrow S \twoheadrightarrow T'$ implies $T \twoheadrightarrow S' \twoheadleftarrow T'$ for some infinite term S'.

The following lemma and theorem are well-known:

Lemma 7.1.4 (Strip Lemma). *Let $\mathcal{R} = (\Sigma, R)$ be an orthogonal iTRS and let $S, T_1, T_2 \in \mathcal{T}er^\infty(\Sigma, V)$. If $T_1 \to S \twoheadleftarrow T_2$, then there exist $S' \in \mathcal{T}er^\infty(\Sigma, V)$ such that $T_1 \twoheadrightarrow S' \twoheadleftarrow T_2$ (see Figure 7.1).*

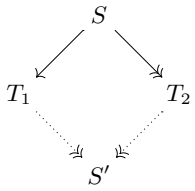

Figure 7.1. The Strip Lemma

Proof. This is Lemma 4.8 of [KKSV95] and Theorem 12.6.3 of [Ter03]. □

Theorem 7.1.5 (Compression). *Let $\mathcal{R} = (\Sigma, R)$ be a left-linear iTRS and let $S, T \in \mathcal{T}er^\infty(\Sigma, V)$. If $S \twoheadrightarrow^\alpha T$, then $S \twoheadrightarrow^{\leqslant\omega} T$.*

Proof. This is Lemma 5.1 of [KKSV95] and Theorem 12.7.1 of [Ter03]. □

Define an infinite term S to be *root-stable* whenever there does not exist a strongly convergent reduction from S to an infinite term with a redex at the root. We have the following:

Corollary 7.1.6. *Let $\mathcal{R} = (\Sigma, R)$ be a left-linear iTRS and let $S \in \mathcal{T}er^\infty(\Sigma, V)$. If S is not root-stable, then it is reduces to a redex in a finite number of steps. Moreover, if $\mathcal{R}$ is orthogonal and S reduces to a root-stable term, then it does so in a finite number of steps.*

Proof. Immediate by compression and strong convergence. □

7.2 Böhm-Like Trees and Infinitary Rewriting

Defining Böhm-like trees for TRSs by means of infinitary rewriting is done in four steps. The steps are similar to the steps employed to define the Böhm-like trees of the $\lambda\beta$-calculus by means of infinitary rewriting (see Section 4.4.2). They are as follows:

1. Define a set $\mathcal{U} \subseteq \mathcal{T}er^{\infty}(\Sigma_{\perp}, V)$.
2. Introduce $B = \{S \to_{\perp} \perp \mid S \in \mathcal{U} \text{ and } S \neq \perp\}$ and $\mathcal{B} = (\Sigma_{\perp}, R \cup B)$.
3. Show that $\mathcal{B}$ is confluent and normalising.
4. Define Böhm-like trees as the unique normal forms with respect to $\mathcal{B}$.

We briefly discuss each of the above steps. A more elaborate discussion can be found in the paper by Kennaway, Van Oostrom, and De Vries [KOV99].

First Step. In this step, a set $\mathcal{U}$ of infinite terms is defined. Intuitively, we want each term in $\mathcal{U}$ to have the infinite term $\perp$ as its Böhm-like tree.

Not every set of infinite terms will do as the set $\mathcal{U}$, since confluence and normalisation are to be proved in the third step. To achieve confluence and normalisation, Kennaway, Van Oostrom, and De Vries require $\mathcal{U}$ to satisfy the following properties, given that $\perp$ is a fresh nullary function symbol that occurs neither in Σ nor in V:

Definition 7.2.1. *A set $\mathcal{U} \subseteq \mathcal{T}er^{\infty}(\Sigma_{\perp}, V)$ is called a set of* meaningless terms, *whenever the following holds for all $S, T \in \mathcal{T}er^{\infty}(\Sigma_{\perp}, V)$:*

1. *$\perp \in \mathcal{U}$,*
2. *$\mathcal{U}$ is closed under strongly convergent reductions,*
3. *if $S \in \mathcal{U}$ overlaps the left-hand side l of a rewrite rule in $\mathcal{R}$ at a position $p \in \mathcal{P}os(l)$ with σ as mgu, then $\sigma(l) = \sigma(l[S]_p) \in \mathcal{U}$,*
4. *all root-active terms occur in $\mathcal{U}$, and*
5. *if $S \leftrightarrow_{\mathcal{U}} T$, then $S \in \mathcal{U}$ if and only if $T \in \mathcal{U}$.*

Above, $S \leftrightarrow_{\mathcal{U}} T$ denotes that T can be obtained from S by replacing a number of disjoint subterms in S that are in $\mathcal{U}$ by other terms from $\mathcal{U}$. Remark that $\mathcal{U}$ can contain both finite and non-finite terms, because $\mathcal{U}$ is defined in the context of infinitary rewriting.

Sets of meaningless terms are not necessarily closed under expansion. This becomes problematic in the comparison with the direct approximant approach, since the direct approximant approach requires $\omega(s) \preccurlyeq \omega(t)$ whenever $s \to t$.

To see that closure under expansion is not necessarily satisfied, consider the orthogonal iTRS which consist solely of the following rewrite rule:

$$a \to b\,.$$

With respect to the iTRS, it is readily proved that $\mathcal{U} = \{\perp, b\}$ is a set of meaningless terms. However, the set is not closed under expansion, since $a \notin \mathcal{U}$.

To overcome the problem with expansion, we also consider the following set:

Definition 7.2.2. *The set $\overline{\mathcal{U}} \subseteq \mathcal{T}er^{\infty}(\Sigma_{\perp}, V)$ is the closure of $\mathcal{U}$ under* (transfinite) expansion. *That is, $\overline{\mathcal{U}}$ is defined as:*

$$\overline{\mathcal{U}} = \{S \mid S \twoheadrightarrow T \text{ with } T \in \mathcal{U}\}\,.$$

Remark that $\mathcal{U} \subseteq \overline{\mathcal{U}}$. The sets may be equal, as $\mathcal{U}$ may already be closed under expansion. The set $\overline{\mathcal{U}}$ does not occur in the work of Kennaway, Van Oostrom, and De Vries [KOV99].

Second Step. Assuming from here onwards that $\mathcal{U}$ is a set of meaningless terms, the current step defines the following iTRS for $\mathcal{R}$ and $\mathcal{U}$:

Definition 7.2.3. *The* Böhm-like iTRS *for $\mathcal{R}$ and $\mathcal{U}$ is defined as the iTRS $\mathcal{B} = (\Sigma_\bot, R \cup B)$, where:*

$$B = \{S \to_\bot \bot \mid S \in \mathcal{U} \text{ and } S \neq \bot\}.$$

With respect to the rewrite rules of B, we have the following compression property, where $\twoheadrightarrow_\bot$ denotes that all employed rewrite rules are from B:

Lemma 7.2.4. *Let $\mathcal{B} = (\Sigma_\bot, R \cup B)$ be a Böhm-like iTRS and $S, T \in \mathcal{T}er^\infty(\Sigma_\bot, V)$. If $S \twoheadrightarrow_\bot T$, then $S \twoheadrightarrow_\bot^{\leqslant\omega} T$.*

Proof. Suppose $S \twoheadrightarrow_\bot T$. By the first and fifth clause of Definition 7.2.1 and the definition of Böhm-like iTRSs, we can assume that all reductions in $S \twoheadrightarrow_\bot T$ occur at parallel positions. Hence, all contracted redexes occur in S. Since there are only finitely many positions at each depth, we obtain a strongly convergent reduction sequence by contracting the redexes in S that need to be contracted in a depth-wise fashion starting at the least depth. By definition, the constructed reduction has length at most ω. That the constructed reduction ends in T follows by definition of Böhm-like iTRSs, in particular by the fact that all right-hand sides of rewrite rules in B are equal to $\bot$. □

Besides the above compression property, we also have the following, where $\twoheadrightarrow_{\mathcal{R}}$ denotes that all employed rewrite rules are from R:

Lemma 7.2.5. *Let $\mathcal{B} = (\Sigma_\bot, R \cup B)$ be a Böhm-like iTRS and $S, T \in \mathcal{T}er^\infty(\Sigma_\bot, V)$. If $S \twoheadrightarrow T$, then $S \twoheadrightarrow_{\mathcal{R}}^{\leqslant\omega} S' \twoheadrightarrow_\bot^{\leqslant\omega} T$ for some $S' \in \mathcal{T}er^\infty(\Sigma_\bot, V)$.*

Proof. This is Lemma 27 of [KOV99] combined with Theorem 7.1.5 and Lemma 7.2.4. □

The above definition of a Böhm-like iTRS is based on $\mathcal{U}$ and not $\overline{\mathcal{U}}$. However, we can also define:

Definition 7.2.6. *The* expanded Böhm-like iTRS *for $\mathcal{R}$ and $\mathcal{U}$ is defined as the iTRS $\mathcal{E} = (\Sigma_\bot, R \cup E)$, where:*

$$E = \{S \to_\bot \bot \mid S \in \overline{\mathcal{U}} \text{ and } S \neq \bot\}.$$

We have the following correspondence between strongly convergent reductions of Böhm-like iTRSs and expanded Böhm-like iTRSs:

Proposition 7.2.7. *Let $\mathcal{B} = (\Sigma_\bot, R \cup B)$ and $\mathcal{E} = (\Sigma_\bot, R \cup B)$ be respectively the Böhm-like iTRS and the expanded Böhm-like iTRS for $\mathcal{R}$ and $\mathcal{U}$. Moreover, let $S, T \in \mathcal{T}er^\infty(\Sigma_\bot, V)$. There exists a reduction $S \twoheadrightarrow T$ with respect to $\mathcal{B}$ if and only if there exists a reduction $S \twoheadrightarrow T$ with respect to $\mathcal{E}$.*

Proof. Let $S \twoheadrightarrow T$ with respect to $\mathcal{B}$. That there exists a reduction $S \twoheadrightarrow T$ with respect to $\mathcal{E}$ is immediate by $R \cup B \subseteq R \cup E$, which follows by $\mathcal{U} \subseteq \overline{\mathcal{U}}$.

Let $S \twoheadrightarrow T$ with respect to $\mathcal{E}$. Remark that the rewrite rules that are in $R \cup E$ but not in $R \cup B$ are of the form $S \to_\perp \perp$ with $S \in \overline{\mathcal{U}} - \mathcal{U}$. Moreover, if $S \to_\perp \perp$ is such a rule, then there exists by definition of $\overline{\mathcal{U}}$ and Theorem 7.1.5 a reduction $S \twoheadrightarrow_{\mathcal{R}}^{\leqslant\omega} T \to_\perp \perp$ for some $T \in \mathcal{U}$ with respect to $\mathcal{B}$. Hence, we can replace each of the reduction steps $C[S] \to_\perp C[\perp]$ in $S \twoheadrightarrow T$ employing one of the 'offending' reduction rules by $C[S] \twoheadrightarrow_{\mathcal{R}}^{\leqslant\omega} C[T] \to_\perp C[\perp]$. The newly created reduction is strongly convergent, as the substituted reductions have at most length $\omega + 1$ and as all reduction steps in $C[S] \twoheadrightarrow_{\mathcal{R}}^{\leqslant\omega} C[T] \to_\perp C[\perp]$ occur at depths greater than or equal to the depth of the hole in the context. □

We now also have the following:

Lemma 7.2.8. *Let $\mathcal{E} = (\Sigma_\perp, R \cup E)$ be an expanded Böhm-like iTRS and let $S, T \in \mathcal{T}er^\infty(\Sigma_\perp, V)$. If $S \twoheadrightarrow T$, then $S \twoheadrightarrow^{\leqslant\omega} T$.*

Proof. Let $S \twoheadrightarrow T$. By Proposition 7.2.7 and Lemma 7.2.5 there exists with respect to $\mathcal{E}$ a reduction $S \twoheadrightarrow_{\mathcal{R}}^{\leqslant\omega} S' \twoheadrightarrow_\perp^{\leqslant\omega} T$ with $S' \in \mathcal{T}er^\infty(\Sigma_\perp, V)$. By closure of $\overline{\mathcal{U}}$ under expansion and definition of E there exists an interleaving of the steps from $S \twoheadrightarrow_{\mathcal{R}}^{\leqslant\omega} S'$ and those from $S' \twoheadrightarrow_\perp^{\leqslant\omega} T$ where the steps occur in a depth-wise fashion starting at the least depth. This implies $S \twoheadrightarrow^{\leqslant\omega} T$, as required. □

Third Step. This step consists of proving the following theorem:

Theorem 7.2.9. *Each Böhm-like iTRS is confluent and normalising.*

Proof. This is Section 7 of [KOV99]. □

Obviously, by the above theorem, each term has a unique normal form with respect to a Böhm-like iTRS.

We have an identical theorem for expanded Böhm-like iTRSs:

Theorem 7.2.10. *Each expanded Böhm-like iTRS is confluent and normalising.*

Proof. Immediate by Theorem 7.2.9 and Proposition 7.2.7. □

Hence, each term also has a unique normal form with respect to an expanded Böhm-like iTRS. By Proposition 7.2.7, the unique normal form must be identical to one with respect to the Böhm-like iTRS.

We also have the following:

Lemma 7.2.11. *The set $\overline{\mathcal{U}}$ is closed under strongly convergent reductions.*

Proof. Suppose $S \in \overline{\mathcal{U}}$ and $S \twoheadrightarrow T$ such that $T \notin \overline{\mathcal{U}}$. By definition of $\overline{\mathcal{U}}$ there exist $S' \in \mathcal{U}$ such that $S \twoheadrightarrow S'$. We also have that there exist $T' \in \mathcal{U}$ such that $T \twoheadrightarrow T'$. If not, then we have found a counterexample to confluence of Böhm-like iTRSs, as $S \twoheadrightarrow S' \twoheadrightarrow \perp$ and $S \twoheadrightarrow T \not\twoheadrightarrow \perp$. Hence, as $T' \in \mathcal{U}$, we have by closure of $\overline{\mathcal{U}}$ under expansion that $T \in \overline{\mathcal{U}}$. □

Fourth Step. The *Böhm-like tree* of a term $S \in \mathcal{T}er^\infty(\Sigma_\perp, V)$ with respect to $\mathcal{U}$, denoted either $\mathrm{BLT}_{\mathcal{U}}^\infty(S)$ or $\mathrm{BLT}^\infty(S)$, is defined as the unique normal form of S with respect to the Böhm-like iTRS for $\mathcal{U}$ and the assumed iTRS.

Alternatively, the Böhm-like tree of a term can be defined as the unique normal form with respect to the expanded Böhm-like iTRS. This makes no difference, because the unique normal forms with respect to both iTRSs are identical, as we saw above. Given this fact, we assume in the remainder of this chapter that Böhm-like trees are defined by means of *expanded* Böhm-like iTRSs.

We have the following for the Böhm-like trees we just defined:

Theorem 7.2.12. *Let $S, T \in \mathcal{T}er^{\infty}(\Sigma, V)$ and let $C[\Box]$ be a context. If it holds that $\mathrm{BLT}^{\infty}(S) = \mathrm{BLT}^{\infty}(T)$, then $\mathrm{BLT}^{\infty}(C[S]) = \mathrm{BLT}^{\infty}(C[T])$.*

Proof. Immediate by confluence of (expanded) Böhm-like iTRSs. □

7.3 From Infinitary Rewriting to Direct Approximants

In this section, we show that each Böhm-like tree defined by means of infinitary rewriting can also be defined by means of a direct approximant function. In other words, we show for each set of meaningless terms that there exists a direct approximant function such that the Böhm-like trees defined by both are identical.

We do not define an ωTRS for each set of meaningless terms. In fact, this is impossible, because Berarducci-like trees can be defined by means of a set of meaningless terms, as we show in Section 7.4.2, while they cannot be defined by means of an ωTRS (see Chapter 5).

Assuming that $\mathcal{R} = (\Sigma, R)$ is an orthogonal TRS and that $\mathcal{U}$ is a set of meaningless terms, we define the following direct approximant function for $\mathcal{U}$:

Definition 7.3.1. *The map $\omega_{\mathcal{U}} : \mathcal{T}er(\Sigma_{\bot}, V) \to \mathcal{T}er(\Sigma_{\bot}, V)$ is defined for each $s \in \mathcal{T}er(\Sigma_{\bot}, V)$ as the largest term $t \in \mathcal{T}er(\Sigma_{\bot}, V)$ with respect to the prefix order such that $t \preccurlyeq s[\bot]_p$ for all $p \in \mathcal{P}os(s)$ with either $s|_p \in \overline{\mathcal{U}}$ or $s|_p$ reducible to a redex of $\mathcal{R}$.*

By Corollary 7.1.6, only finite reductions need to be considered to determine whether a subterm reduces to a redex in $\mathcal{R}$. Of course, this does not make the requirement decidable in any way.

Remark 7.3.2. It is possible to give an alternative definition for $\omega_{\mathcal{U}}$, which employs the expanded Böhm-like iTRS $\mathcal{E}$ for $\mathcal{U}$ and the prefix order on positions instead of the set $\overline{\mathcal{U}}$, the TRS $\mathcal{R}$, and the prefix order on terms.

Suppose $s \in \mathcal{T}er(\Sigma_{\bot}, V)$ and define:

$$\mathcal{P}_s = \{p \mid p \in \mathcal{P}os(s) \text{ and } s|_p \text{ reducible to a redex of } \mathcal{E}\}$$
$$\mathcal{P}_s^{\mathrm{o}} = \{p \mid p \in \mathcal{P}_s \text{ and no } q < p \text{ in } \mathcal{P}_s\}$$

where $\mathcal{P}_s^{\mathrm{o}}$ consists of the outermost positions of $\mathcal{P}_s$. The value assigned to $\omega_{\mathcal{U}}(s)$ can now be defined as:

$$\omega_{\mathcal{U}}(s) = s[\bot]_{p_1}[\bot]_{p_2} \ldots [\bot]_{p_n} ,$$

where $\#\mathcal{P}_s^{\mathrm{o}} = n$ and where $p_1, \ldots, p_n$ are the different (parallel) positions that occur in $\mathcal{P}_s^{\mathrm{o}}$.

By definition of $\mathcal{E}$ and closure of $\overline{\mathcal{U}}$ under expansion, it follows that reducibility to a redex of $\mathcal{E}$ corresponds to either reducibility to a redex of $\mathcal{R}$ or equality to a term in $\overline{\mathcal{U}}$. Hence, as the term t from Definition 7.3.1 is the largest term with respect to the prefix order and as $\mathcal{P}^{\mathrm{o}}_s$ consists of the outermost positions from $\mathcal{P}_s$, we have that $s[\bot]_{p_1}[\bot]_{p_2}\ldots[\bot]_{p_n}$ is identical to t.

Of course, we still have to show the following:

Lemma 7.3.3. *The map $\omega_{\mathcal{U}}$ is a direct approximant function.*

Proof. As $\mathcal{R}$ is assumed to be orthogonal, it is sufficient by Proposition 5.2.14 to verify the first three clauses of the definition of a direct approximant function. Assuming $s,t \in \mathcal{T}er(\Sigma_\bot, V)$, we verify each of the three clauses in turn:

1. That $\omega_{\mathcal{U}}(s) \preccurlyeq s$, is immediate by the alternative definition of $\omega_{\mathcal{U}}$ from Remark 7.3.2.
2. That $\omega_{\mathcal{U}}(s) \preccurlyeq s[\bot]_p$ for all $p \in \mathcal{P}os(s)$ such that $s|_p = \sigma(l)$ for some $l \to r \in R$ and substitution σ, is immediate by the requirement that $\omega_{\mathcal{U}}(s) \preccurlyeq s[\bot]_q$ whenever $s|_q$ is reducible to a redex of $\mathcal{R}$.
3. That $s \to t$ implies $\omega_{\mathcal{U}}(s) \preccurlyeq \omega_{\mathcal{U}}(t)$, is immediate whenever there exists for each $p \in \mathcal{P}^{\mathrm{o}}_t$ a position $p' \leqslant p$ such that $\omega_{\mathcal{U}}(s) \preccurlyeq s[\bot]_{p'}$. To show this, suppose that q is the position of the redex contracted in $s \to t$. There are three possibilities depending of the relative positions of p and q.
   If $p \parallel q$, then $p \in \mathcal{P}os(s)$ and either $s|_p \in \overline{\mathcal{U}}$ or $s|_p$ reduces to a redex of $\mathcal{R}$. Hence, $\omega_{\mathcal{U}}(s) \preccurlyeq s[\bot]_p$. If $q \leqslant p$, then, since a redex occurs at position q in s, we have $\omega_{\mathcal{U}}(s) \preccurlyeq s[\bot]_q$. Finally, if $p < q$, then there are two possibilities: either $t|_p$ reduces to a redex of $\mathcal{R}$ or $t|_p \in \overline{\mathcal{U}}$. If $t|_p$ reduces to a redex, then $s|_p$ also reduces to a redex, since s reduces to t. If $t|_p \in \overline{\mathcal{U}}$, then $s|_p \in \overline{\mathcal{U}}$, because $\overline{\mathcal{U}}$ is closed under expansion. Thus, in both cases $\omega_{\mathcal{U}}(s) \preccurlyeq s[\bot]_p$.
   Hence, for each $p \in \mathcal{P}^{\mathrm{o}}_t$ there exists a position $p' \leqslant p$ such that $\omega_{\mathcal{U}}(s) \preccurlyeq s[\bot]_{p'}$, as required for the third clause. □

Assume that the Böhm-like trees based on the expanded Böhm-like iTRS $\mathcal{E}$ of $\mathcal{U}$ and those based on $\omega_{\mathcal{U}}$ are denoted respectively $\mathrm{BLT}^{\infty}_{\mathcal{U}}$ and $\mathrm{BLT}_{\mathcal{U}}$, we next show that $\mathrm{BLT}^{\infty}_{\mathcal{U}}$ and $\mathrm{BLT}_{\mathcal{U}}$ are identical. The proof essentially consists of the construction of a bisimulation:

Theorem 7.3.4. *For all $s \in \mathcal{T}er(\Sigma_\bot, V)$ it holds that $\mathrm{BLT}_{\mathcal{U}}(s) = \mathrm{BLT}^{\infty}_{\mathcal{U}}(s)$.*

Proof. Let $s \in \mathcal{T}er(\Sigma_\bot, V)$. We prove by induction on the structure of positions $p \in \mathbb{N}^*$ that $p \in \mathcal{P}os(\mathrm{BLT}_{\mathcal{U}}(s))$ if and only if $p \in \mathcal{P}os(\mathrm{BLT}^{\infty}_{\mathcal{U}}(s))$ and that $root(\mathrm{BLT}_{\mathcal{U}}(s)|_p) = root(\mathrm{BLT}^{\infty}_{\mathcal{U}}(s)|_p)$.

Obviously, if $p = \epsilon$, then p is in the set of positions of both Böhm-like trees. Otherwise, if $p = q \cdot i$ for some $i \in \mathbb{N}$, then p is in the set of positions of both Böhm-like trees whenever q is in the set of positions and whenever $root(\mathrm{BLT}_{\mathcal{U}}(s)|_q) = root(\mathrm{BLT}^{\infty}_{\mathcal{U}}(s)|_q) \in \Sigma_n$ with $0 \leqslant i \leqslant n$. Hence, we only need to prove for each position p in both Böhm-like trees that $root(\mathrm{BLT}_{\mathcal{U}}(s)|_p) = root(\mathrm{BLT}^{\infty}_{\mathcal{U}}(s)|_p)$.

Suppose $root(\mathrm{BLT}_{\mathcal{U}}(s)|_p) = f$. There are two possibilities: Either $f = \bot$ or $f \neq \bot$. In case $f = \bot$, we have, by definition of $\omega_{\mathcal{U}}$ for all $s \to^* t$ with $p \in \mathcal{P}os(\omega_{\mathcal{U}}(t))$,

that either $t|_p \in \overline{\mathcal{U}}$ or that $t|_p$ reduces to a redex of $\mathcal{R}$, which, by Corollary 7.1.6 and the fourth clause of Definition 7.2.1, also implies that $t|_p \in \overline{\mathcal{U}}$. Hence, in any case $t|_p \in \overline{\mathcal{U}}$ and we obtain $root(\mathrm{BLT}^{\infty}_{\mathcal{U}}(s)|_p) = \bot$.

In case $f \neq \bot$, there exists by definition of $\omega_{\mathcal{U}}$ a term t such that $s \to^* t$, $p \in \mathcal{P}os(\omega_{\mathcal{U}}(t))$, and $root(\omega_{\mathcal{U}}(t)|_p) = f$. Hence, by definition of $\omega_{\mathcal{U}}$, we have neither that $t|_p \in \overline{\mathcal{U}}$ nor that $t|_p$ reduces to a redex of $\mathcal{R}$, which implies $root(\mathrm{BLT}^{\infty}_{\mathcal{U}}(s)|_p) = f$.

Now suppose $root(\mathrm{BLT}^{\infty}_{\mathcal{U}}(s)|_p) = f$. There are again two possibilities: Either $f = \bot$ or $f \neq \bot$. In case $f = \bot$, we have, by Lemma 7.2.8 and the definition of Böhm-like trees based on infinitary rewriting, that there exist $s \to^* t$ such that $p \in \mathcal{P}os(t)$, all $t|_q$ with $q < p$ root-stable with respect to $\mathcal{E}$, and $t|_p \in \overline{\mathcal{U}}$. Hence, by definition of $\omega_{\mathcal{U}}$ and Lemma 7.2.11, we have $p \in \mathcal{P}os(\omega_{\mathcal{U}}(t))$ and $\omega_{\mathcal{U}}(t)|_p = \bot$, which implies $root(\mathrm{BLT}_{\mathcal{U}}(s)|_p) = \bot$.

In case $f \neq \bot$, there exists, by definition of Böhm-like trees based on infinitary rewriting and Corollary 7.1.6, a term t such that $t|_q$ with $q < p$ root-stable with respect to $\mathcal{E}$. Hence, $root(\omega_{\mathcal{U}}(t)) = f$, which implies $root(\mathrm{BLT}_{\mathcal{U}}(s)|_p) = f$.

Concluding, we have $root(\mathrm{BLT}_{\mathcal{U}}(s)|_p) = f$ if and only if $root(\mathrm{BLT}^{\infty}_{\mathcal{U}}(s)|_p) = f$, as required. □

Discussion. Given the direct approximant function $\omega_{\mathcal{U}}$ based on the set $\mathcal{U}$ of meaningless terms, we can ask ourselves whether we can 'recover' the set $\overline{\mathcal{U}}$. In this respect, we have the following:

Lemma 7.3.5. *Define the set* $\mathcal{U}' \subseteq \mathcal{T}er(\Sigma_\bot, V)$ *as:*

$$\mathcal{U}' = \{s \mid \forall s \to^* t : \omega_{\mathcal{U}}(t) = \bot\}\,.$$

The set $\mathcal{U}'$ *consists of all finite terms from* $\overline{\mathcal{U}}$.

Proof. Denote by $\overline{\mathcal{U}}^{\mathrm{f}}$ the subset of $\overline{\mathcal{U}}$ that consists of all finite terms. We prove $\overline{\mathcal{U}}^{\mathrm{f}} \subseteq \mathcal{U}'$ and $\mathcal{U}' \subseteq \overline{\mathcal{U}}^{\mathrm{f}}$, by which the result is immediate.

To prove $\overline{\mathcal{U}}^{\mathrm{f}} \subseteq \mathcal{U}'$, suppose $s \in \overline{\mathcal{U}}^{\mathrm{f}}$. As $\overline{\mathcal{U}}^{\mathrm{f}} \subseteq \overline{\mathcal{U}}$ and as $\overline{\mathcal{U}}$ is closed under reduction, we have for all $s \to^* t$ that $t \in \overline{\mathcal{U}}$. Hence, $s \in \mathcal{U}'$. As s was arbitrary, we can conclude that $\overline{\mathcal{U}}^{\mathrm{f}} \subseteq \mathcal{U}'$.

To prove $\mathcal{U}' \subseteq \overline{\mathcal{U}}^{\mathrm{f}}$, suppose $s \in \mathcal{U}'$. By definition of $\omega_{\mathcal{U}}$ there are two possibilities: Either there exist $s \to^* t$ such that $t \in \overline{\mathcal{U}}$ or for all $s \to^* t$ we have that t is reducible to a redex of $\mathcal{R}$. In the first case, by closure of $\overline{\mathcal{U}}$ under expansion, we have that $s \in \overline{\mathcal{U}}^{\mathrm{f}}$. In the second case, s is root-active, which, by Corollary 7.1.6, also implies $s \in \overline{\mathcal{U}}^{\mathrm{f}}$. As s was arbitrary, we can conclude that $\mathcal{U}' \subseteq \overline{\mathcal{U}}^{\mathrm{f}}$. □

Although we just recovered the finite terms of $\overline{\mathcal{U}}$, we cannot recover $\overline{\mathcal{U}}$ completely. To see this, consider a TRS without any rewrite rules whose signature consists of the unary function symbols f and g. We can define the following infinite term:

$$S = f(g(f(g^2(f(g^3(f(\ldots)))))))\,.$$

That is, each subterm of S with f as root symbol is of the form:

$$f(g^n(f(g^{n+1}(\ldots))))\,.$$

Since there are no reduction rules, both $\mathcal{U} = \{\bot\}$ and $\mathcal{V} = \{\bot, S\}$ are sets of meaningless terms. Moreover, $\overline{\mathcal{U}} = \mathcal{U}$ and $\overline{\mathcal{V}} = \mathcal{V}$. Since S is an infinite term, both $\omega_{\mathcal{U}}$ and $\omega_{\mathcal{V}}$ can only be the identity function. Hence, it is impossible to recover either $\mathcal{U}$ or $\mathcal{V}$.

Even though the above shows that we cannot recover all infinite terms from $\overline{\mathcal{U}}$, we might be able to recover some of the non-finite terms. A possible way of doing this is suggested by the paper of Kennaway, Van Oostrom, and De Vries [KOV99] (see also the next section). In the paper, some sets of meaningless terms are defined in two steps. In the first step, a direct approximant function ω is defined together with a set collecting the terms all whose reducts have $\bot$ as their direct approximant:

$$\mathcal{U}^{\mathrm{f}} = \{s \mid \forall s \to^* t : \omega(t) = \bot\}\,.$$

Remark that the set is identical to the one defined in Lemma 7.3.5 but with $\omega_{\mathcal{U}}$ replaced by ω. In the second step, the following set is defined:

$$\mathcal{U} = \{S \mid \forall S \twoheadrightarrow T : \forall t \in \mathcal{T}er(\Sigma_\bot, V) : t \preccurlyeq T \Longrightarrow t \in \mathcal{U}^{\mathrm{f}}\}\,.$$

Hence, a term S is meaningless if and only if the all the finite prefixes of all reducts of S are in $\mathcal{U}^{\mathrm{f}}$.

It might be suspected that the two-step construction could be used to recover a subset of the non-finite terms from $\overline{\mathcal{U}}$. However, this is not the case. Consider, e.g., the TRS which only has the following rewrite rule:

$$a \to a\,,$$

In addition, assume that the TRS has a unary function symbol f. In this case, the set:

$$\mathcal{U} = \{f^n(a), f^n(\bot) \mid n \in \mathbb{N}\}\,,$$

with $f^0(s) = s$ and $f^{n+1}(s) = f(f^n(s))$, is easily shown to be a set of meaningless terms with the property that $\overline{\mathcal{U}}^{\mathrm{f}} = \overline{\mathcal{U}} = \mathcal{U}$. Unfortunately, besides $\mathcal{U}$ being a subset of the following set:

$$\{S \mid \forall S \twoheadrightarrow T : \forall s \in \mathcal{T}er(\Sigma_\bot, V) : s \preccurlyeq T \Longrightarrow s \in \mathcal{U}\}\,,$$

f^ω is also in the set, since $f^n(\bot) \in \mathcal{U}$ for all $n \in \mathbb{N}$. Thus, we not only recover all non-finite terms from $\overline{\mathcal{U}}$, but we also 'recover' terms that were not in $\overline{\mathcal{U}}$. Hence, it is probable that all we can hope for is Lemma 7.3.5.

7.4 From Direct Approximants to Infinitary Rewriting

Without posing any restrictions on Böhm-like trees based on direct approximant functions, we cannot prove the reverse of the theorem established in the previous section. That is, we cannot prove for every Böhm-like tree based on a direct approximant function that there exists an identical Böhm-like tree defined by means of infinitary rewriting. It is impossible to give a proof, since Böhm-like trees based

on infinitary rewriting satisfy congruence (see Theorem 7.2.12), while congruence is not necessarily satisfied by Böhm-like trees based on direct approximant functions (see Chapter 6).

Remark that it is irrelevant whether the direct approximant function is based on an ωTRS. Congruence is not necessarily satisfied by Böhm-like trees based on ωTRSs either (see also Chapter 6).

As it turns out, it is not sufficient to just assume congruence for Böhm-like trees based on direct approximant functions, as we show in Section 7.4.1. However, as shown in Sections 7.4.2, 7.4.3, and 7.4.4, we can prove that there exist Böhm-like trees based on infinitary rewriting in case of Berarducci-like trees, Huet-Lévy trees, and the trees based on ω_{m}TRSs.

7.4.1 Congruence

In this section, we define a Böhm-like tree based on a direct approximant function which satisfies congruence, but for which no identical Böhm-like tree based on infinitary rewriting exists.

Consider the orthogonal TRS which only has the following rewrite rule:

$$f(g(x)) \to f(x)\,.$$

Define the following map:

Definition 7.4.1. *The map $\omega_{\mathrm{c}} : \mathcal{T}er(\Sigma_\perp, V) \to \mathcal{T}er(\Sigma_\perp, V)$ is defined for each $s \in \mathcal{T}er(\Sigma_\perp, V)$ as the largest term $t \in \mathcal{T}er(\Sigma_\perp, V)$ with respect to the prefix order such that $t \preccurlyeq s[\perp]_p$ if either $s|_p$ is a redex or $s|_p = g^n(\perp)$ for some $n \in \mathbb{N}$.*

As we show below, the above map defines a direct approximant function whose Böhm-like tree satisfies congruence. However, we first show that no Böhm-like tree based on infinitary rewriting can be defined.

Denote by BLT the Böhm-tree based on ω_{c} and consider the term $g(\perp)$. Obviously, we have $\mathrm{BLT}(g(\perp)) = \perp$. Since we want to define a set of meaningless terms whose Böhm-like iTRS yields a Böhm-like tree identical to the one based on ω_{c}, it must hold that $g(\perp)$ is in the set of meaningless terms. Alternatively, $g(\perp)$ has to reduce to a term in the set, but this is impossible, since $g(\perp)$ is a normal form with respect to the assumed TRS.

Since $g(\perp)$ overlaps $f(g(x))$, we have by the second clause of Definition 7.2.1 that $f(g(\perp))$ must also be in the set of meaningless terms. This implies, by definition of Böhm-like iTRSs, that $f(g(\perp)) \to_\perp \perp$ and that $\mathrm{BLT}^\infty(f(g(\perp))) = \perp$. However, $\mathrm{BLT}(f(g(\perp))) = \mathrm{BLT}(f(\perp)) = f(\perp)$. Hence, even requiring congruence to hold for Böhm-like trees based on direct approximant functions is insufficient to be able to define an identical Böhm-like tree based on infinitary rewriting.

Of course, we still have to show that ω_{c} defines a direct approximant function that yields a Böhm-like tree which satisfies congruence. We start by proving that ω_{c} is a direct approximant function:

Lemma 7.4.2. *The map ω_{c} is a direct approximant function.*

Proof. Since the considered TRS is orthogonal, it suffices by Proposition 5.2.14 to verify the first three clauses of the definition of a direct approximant function. Assuming $s, t \in \mathcal{T}er(\Sigma_\perp, V)$, we verify each of the three clauses in turn:

1. That $\omega_{\mathrm{c}}(s) \preccurlyeq s$, is immediate by the definition of ω_{c}.
2. That $\omega_{\mathrm{c}}(s) \preccurlyeq s[\perp]_p$ whenever $p \in \mathcal{P}os(s)$ and $s|_p = \sigma(f(g(x)))$ with σ is a substitution, is also immediate by the definition of ω_{c}.
3. That $s \to t$ implies $\omega_{\mathrm{c}}(s) \preccurlyeq \omega_{\mathrm{c}}(t)$, follows immediately when we realise that each subterm of the form $f(g(s'))$ reduces to $f(s')$ and that any redex that is created at the same position as $f(g(s'))$. □

Remark 7.4.3. The replacement of subterms of the form $g^n(\perp)$ by ω_{c} has a similar justification as the rule $\lambda x.\perp \to_\omega \perp$, which occurs in the rewrite system that defines the Böhm direct approximant of the $\lambda\beta$-calculus (see Section 5.4). Placing $g^n(\perp)$ in a context either leaves $g^n(\perp)$ untouched or it creates the redex $f(g^n(\perp))$. In the last case we have $f(g^n(\perp)) \to^* f(\perp) = \omega_{\mathrm{c}}(f(\omega_{\mathrm{c}}(g^n(\perp))))$, which is similar to $(\lambda x.\perp)s \to_\beta \perp = \omega_{\mathrm{B}}(\omega_{\mathrm{B}}(\lambda x.\perp)s)$, as satisfied by the Böhm direct approximant.

Finally, we prove congruence:

Lemma 7.4.4. *The Böhm-like tree based on ω_{c} satisfies congruence.*

Proof. To prove that the Böhm-like tree based on ω_{c} is congruent, remark that each term has a normal form, as contracting a redex reduces the number of function symbols g in a term. Moreover, remark that the assumed TRS is confluent, since it is orthogonal.

By confluence, we may rewrite a term $C[s]$ to normal form by first rewriting s and $C[\Box]$ to normal form and by only contracting thereafter the redexes that occur on the 'boundary' between s and $C[\Box]$. Hence, we only need to verify congruence with respect to the possible normal forms of s and $C[\Box]$.

Each term s reduces to one of four normal forms, with $m \in \mathbb{N}$ and $n \geqslant 1$ arbitrary:

$$\begin{array}{cc} g^m(f^n(x)) & g^m(f^n(\perp)) \\ g^m(x) & g^m(\perp) \end{array}$$

The normal forms correspond to the following Böhm-like trees:

$$\begin{array}{cc} \mathrm{BLT}(g^m(f^n(x))) = {\downarrow}\{g^m(f^n(x))\} & \mathrm{BLT}(g^m(f^n(\perp))) = {\downarrow}\{g^m(f^n(\perp))\} \\ \mathrm{BLT}(g^m(x)) = {\downarrow}\{g^m(x)\} & \mathrm{BLT}(g^m(\perp)) = \{\perp\} \end{array}$$

Hence, any two terms have the same Böhm-like trees if they have the same normal form unequal to $g^m(\perp)$ or if the terms both have a normal form $g^m(\perp)$ for possibly different values of m.

Each context $C[\Box]$ reduces to one of the following normal forms, with $m' \in \mathbb{N}$ and $n' \geqslant 1$ arbitrary:

$$g^{m'}(f^{n'}(\Box)) \qquad g^{m'}(\Box)$$

There are eight possible combinations of the normal forms of s and the normal forms of $C[\Box]$. In case $s \neq g^m(\bot)$, the Böhm-like trees are as follows:

$$\begin{aligned}
\mathrm{BLT}(g^{m'}(f^{n'}(g^m(f^n(x))))) &= \downarrow\{g^{m'}(f^{n'+n}(x))\} \\
\mathrm{BLT}(g^{m'}(f^{n'}(g^m(f^n(\bot))))) &= \downarrow\{g^{m'}(f^{n'+n}(\bot))\} \\
\mathrm{BLT}(g^{m'}(f^{n'}(g^m(x)))) &= \downarrow\{g^{m'}(f^{n'}(x))\} \\
\mathrm{BLT}(g^{m'}(g^m(f^n(x)))) &= \downarrow\{g^{m'+m}(f^n(x))\} \\
\mathrm{BLT}(g^{m'}(g^m(f^n(\bot)))) &= \downarrow\{g^{m'+m}(f^n(\bot))\} \\
\mathrm{BLT}(g^{m'}(g^m(x))) &= \downarrow\{g^{m'+m}(x)\}
\end{aligned}$$

Obviously, given two terms with the same normal form unequal to $g^m(\bot)$, we obtain identical Böhm-like trees when we place the terms in a context.

In case $s = g^m(\bot)$, the Böhm-like trees are:

$$\begin{aligned}
\mathrm{BLT}(g^{m'}(f^{n'}(g^m(\bot)))) &= \downarrow\{g^{m'}(f^{n'}(\bot))\} \\
\mathrm{BLT}(g^{m'}(g^m(\bot))) &= \downarrow\{\bot\}
\end{aligned}$$

Hence, the value of m is irrelevant and given any two terms whose normal form is $g^m(\bot)$, for possibly different values of m, we have that their Böhm-like trees are also identical when placed in a context.

Summarising, we have that congruence holds for the Böhm-like tree based on the direct approximant function ω_{c}. □

7.4.2 Berarducci-Like Trees

In this section, we show for every orthogonal TRS $\mathcal{R} = (\Sigma, R)$ and its Berarducci-like tree that there exists a set $\mathcal{U}_{\mathrm{BeL}}$ of meaningless terms such that the Böhm-like tree based on $\mathcal{U}_{\mathrm{BeL}}$ is identical to the Berarducci-like tree.

The set of meaningless terms we are looking for has actually already been defined by Kennaway, Van Oostrom, and De Vries [KOV99]. In Section 8.1.5 of their paper, they define:

$$\mathcal{U}' = \{S \mid S \in \mathcal{T}er^\infty(\Sigma, V) \text{ root-active}\}\,.$$

Subsequently, they define $\mathcal{U}_{\mathrm{BeL}} \subseteq \mathcal{T}er^\infty(\Sigma_\bot, V)$ as the closure of $\mathcal{U}'_\bot = \mathcal{U}' \cup \{\bot\}$ under $\leftrightarrow_{\mathcal{U}'_\bot}$ and they show that $\mathcal{U}_{\mathrm{BeL}}$ is a set of meaningless terms.

Denoting the Böhm-like tree based on $\mathcal{U}_{\mathrm{BeL}}$ by $\mathrm{BLT}^\infty_{\mathrm{BeL}}$ and the Berarducci-like tree by $\mathrm{BLT}_{\mathrm{BeL}}$, we next show that $\mathrm{BLT}^\infty_{\mathrm{BeL}}$ and $\mathrm{BLT}_{\mathrm{BeL}}$ are identical. We start with a lemma:

Lemma 7.4.5. *Let $\mathcal{U} \subseteq \mathcal{T}er^\infty(\Sigma_\bot, V)$ be defined as:*

$$\mathcal{U} = \{S \mid S \text{ either root-active or } S \to^* \bot\}\,.$$

It holds that $\mathcal{U} = \mathcal{U}_{\mathrm{BeL}}$.

Proof. We prove $\mathcal{U}_{\mathrm{BeL}} \subseteq \mathcal{U}$ and $\mathcal{U} \subseteq \mathcal{U}_{\mathrm{BeL}}$. Thus, suppose $S \in \mathcal{U}_{\mathrm{BeL}}$. By definition of $\mathcal{U}_{\mathrm{BeL}}$, we have that S is created out of a term $T \in \mathcal{U}'$ by replacing a number of root-active subterms by $\bot$. The subterms replaced by $\bot$ may or may not contribute to the root-activeness of T. In case they contribute, we have by orthogonality of $\mathcal{R}$ that $S \to^* \bot$. In case they do not contribute, we have by orthogonality that S is root-active. Hence, $S \in \mathcal{U}$.

That $\mathcal{U} \subseteq \mathcal{U}_{\mathrm{BeL}}$ follows by replacing each $\bot$ in every term of $\mathcal{U}$ by a root-active term and by orthogonality of $\mathcal{R}$. If no root-active term exists, then $\mathcal{U} \neq \{\bot\}$ and we are done immediately. □

By the previous lemma and compression, we have that $\mathcal{U}^{\mathrm{BeL}}$ is closed under expansion. That is, we have:

$$\overline{\mathcal{U}}_{\mathrm{BeL}} = \mathcal{U}_{\mathrm{BeL}}\,.$$

We can now prove:

Theorem 7.4.6. *If $s \in \mathcal{T}er(\Sigma_\bot, V)$, then $\mathrm{BLT}^{\infty}_{\mathrm{BeL}}(s) = \mathrm{BLT}_{\mathrm{BeL}}(s)$.*

Proof. Suppose $\omega_{\mathcal{U}_{\mathrm{BeL}}}$ is defined according to Definition 7.3.1, where $\mathcal{U}$ is instantiated by $\mathcal{U}_{\mathrm{BeL}}$. By Lemma 7.4.5, the fact that $\mathcal{U}_{\mathrm{BeL}}$ is closed under expansion, and Definition 7.3.1, we have that $\omega_{\mathcal{U}_{\mathrm{BeL}}}$ replaces precisely every non-root-stable subterm of a term by $\bot$. Hence, $\omega_{\mathcal{U}_{\mathrm{BeL}}} = \omega_{\mathrm{BeL}}$, with ω_{BeL} the Berarducci-like direct approximant defined in Chapter 5. By Theorem 7.3.4, we now have for all $s \in \mathcal{T}er(\Sigma_\bot, V)$ that $\mathrm{BLT}^{\infty}_{\mathrm{BeL}}(s) = \mathrm{BLT}_{\mathrm{BeL}}(s)$. □

We also have the following:

Theorem 7.4.7. *Congruence holds for Berarducci-like trees.*

Proof. Immediate by the Theorems 7.4.6 and 7.2.12. □

7.4.3 Huet-Lévy Trees

In this section, we show for each orthogonal TRS $\mathcal{R} = (\Sigma, R)$ and its Huet-Lévy tree that there exists a set $\mathcal{U}_{\mathrm{HL}}$ of meaningless terms such that the Böhm-like trees based on $\mathcal{U}_{\mathrm{HL}}$ is identical to the Huet-Lévy tree.

As in the case of Berarducci-like trees, we have that the set of meaningless terms has already been defined by Kennaway, Van Oostrom, and De Vries [KOV99]. The definition in Section 8.1.4 of their paper follows the approach in the discussion at the end of Section 7.3. As such, Kennaway, Van Oostrom, and De Vries first define:

$$\mathcal{U}^{\mathrm{f}}_{\mathrm{HL}} = \{s \mid \forall s \to^* t : \omega_{\mathrm{HL}}(t) = \bot\}\,,$$

where ω_{HL} is the Huet-Lévy direct approximant function. Thereafter, they define:

$$\mathcal{U}_{\mathrm{HL}} = \{S \mid \forall S \twoheadrightarrow T : \forall t \in \mathcal{T}er(\Sigma_\bot, V) : t \preccurlyeq T \Longrightarrow t \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}\}\,.$$

Finally, they show that $\mathcal{U}_{\mathrm{HL}}$ is a set of meaningless terms.

Denoting by $\mathrm{BLT}^{\infty}_{\mathrm{HL}}$ the Böhm-like tree based on $\mathcal{U}_{\mathrm{HL}}$ and by $\mathrm{BLT}_{\mathrm{HL}}$ the Huet-Lévy tree, we next show that $\mathrm{BLT}^{\infty}_{\mathrm{HL}}$ and $\mathrm{BLT}_{\mathrm{HL}}$ are identical. To facilitate the proof, we first show that $\mathcal{U}_{\mathrm{HL}}$ is closed under expansion, which requires:

Lemma 7.4.8. *Let $s \in \mathcal{T}er(\Sigma_\bot, V)$ and $S, T \in \mathcal{T}er^\infty(\Sigma_\bot, V)$. If $s \preccurlyeq S$, $S \to^* T$, and $T \in \mathcal{U}_{\mathrm{HL}}$, then $s \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$.*

Proof. Suppose $s \preccurlyeq S$, $S \to^* T$, and $T \in \mathcal{U}_{\mathrm{HL}}$. By definition of $\mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$ we need to show for each $s \to^* s'$ that $\omega_{\mathrm{HL}}(s') = \bot$. Thus, suppose $s \to^* s'$. By orthogonality of $\mathcal{R}$, there exists an infinite term S' such that $s' \preccurlyeq S'$ and $S \to^* S'$. Moreover, by the Strip Lemma and compression, there exists an infinite term T' such that $S' \twoheadrightarrow^{\leqslant \omega} T' \twoheadleftarrow T$. Here, $T' \in \mathcal{U}_{\mathrm{HL}}$, since $\mathcal{U}_{\mathrm{HL}}$ is closed under strongly convergent reductions, as it is a set of meaningless terms.

Since $s' \preccurlyeq S'$ and $S' \twoheadrightarrow T'$, there exist a largest term $t' \preccurlyeq s'$ such that no redex contracted in $S' \twoheadrightarrow T'$ occurs at a position $p \in \mathcal{P}os(t')$ with $t'|_p \neq \bot$. Obviously, $t' \preccurlyeq T'$. Hence, as $T' \in \mathcal{U}_{\mathrm{HL}}$, we have $\omega_{\mathrm{HL}}(t') = \bot$.

By definition of t' it now follows for each $s'|_p \neq \bot$ with $t'|_p = \bot$ that a redex is contracted in $S' \twoheadrightarrow^{\leqslant \omega} T'$ at position p after a finite number of steps. Since the number of steps is finite and since $\mathcal{R}$ is orthogonal, there exist $s'|_p \preccurlyeq s'_p \preccurlyeq S'|_p$ such that s'_p reduces to a redex. Hence, by the second and third clause of the definition of direct approximant functions we have $\omega_{\mathrm{HL}}(s'_p) = \bot$ and, as ω_{HL} can be defined by means of an ωTRS, we have by Lemma 5.4.9.(4) that $\omega_{\mathrm{HL}}(s'|_p) = \bot$. As such, t' can be obtained from s' by replacing each subterm $s'|_p \neq \bot$ by $t'|_p = \bot$ if $\omega_{\mathrm{HL}}(s'|_p) = \bot$. It is now immediate by Lemma 5.4.9.(2) and $\omega_{\mathrm{HL}}(t') = \bot$ that $\omega_{\mathrm{HL}}(s') = \bot$, as required. □

We can now prove that $\mathcal{U}_{\mathrm{HL}}$ is closed under expansion:

Lemma 7.4.9. *It holds that $\overline{\mathcal{U}}_{\mathrm{HL}} = \mathcal{U}_{\mathrm{HL}}$.*

Proof. Let $S \twoheadrightarrow T$ such that $T \in \mathcal{U}_{\mathrm{HL}}$. To prove $S \in \mathcal{U}_{\mathrm{HL}}$, we need to show by definition of $\mathcal{U}_{\mathrm{HL}}$ that for each $S \twoheadrightarrow T'$ and $t' \preccurlyeq T'$ it holds that $t' \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$. Thus, suppose $S \twoheadrightarrow T'$ and $t' \preccurlyeq T'$. By compression and strong convergence, there exists an infinite term T'' such that $S \to^* T''$ and $t' \preccurlyeq T''$. Moreover, by the Strip Lemma we have that there exists an infinite term S' such that $T'' \twoheadrightarrow S' \twoheadleftarrow T$. Obviously, as $\mathcal{U}_{\mathrm{HL}}$ is closed under strongly convergent reductions, we have $S' \in \mathcal{U}_{\mathrm{HL}}$. Hence, the result is now immediate by Lemma 7.4.8. □

We next show that all finite terms that occur in $\mathcal{U}_{\mathrm{HL}}$ already occur in $\mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$. This fact is employed in the proof showing that $\mathrm{BLT}^{\infty}_{\mathrm{HL}}$ and $\mathrm{BLT}_{\mathrm{HL}}$ are identical.

Lemma 7.4.10. *Let $\mathcal{U} \subseteq \mathcal{T}er(\Sigma_\bot, V)$ be defined as:*

$$\mathcal{U} = \{s \mid s \in \mathcal{T}er(\Sigma_\bot, V) \text{ and } s \in \mathcal{U}_{\mathrm{HL}}\}\,.$$

It holds that $\mathcal{U} = \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$.

Proof. We show $\mathcal{U} \subseteq \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$ and $\mathcal{U}^{\mathrm{f}}_{\mathrm{HL}} \subseteq \mathcal{U}$, from which the result follows. Starting with $\mathcal{U} \subseteq \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$, suppose $s \in \mathcal{U}$. Since $s \to^* s$, we have by definition of $\mathcal{U}$ and $\mathcal{U}_{\mathrm{HL}}$ that $s' \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$ for all $s' \preccurlyeq s$. In particular, $s \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$, because $s \preccurlyeq s$.

To show $\mathcal{U}^{\mathrm{f}}_{\mathrm{HL}} \subseteq \mathcal{U}$, suppose $s \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$ and $s \twoheadrightarrow T$. By compression and strong convergence there exists for every $t' \preccurlyeq T$ a term $s' \succcurlyeq t'$ such that $s \to^* s'$. Moreover, by orthogonality of $\mathcal{R}$, there exists for every $t' \to^* t''$ a term $s'' \succcurlyeq t''$ such that

$s' \to^* s''$. Since $s \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$, we have $\omega_{\mathrm{HL}}(s'') = \bot$. Hence, by Lemma 5.4.9.(4), we also have $\omega_{\mathrm{HL}}(t'') = \bot$, which implies $t' \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$, as $t' \to^* t''$ was arbitrary. Since t' was also arbitrary, we obtain $s \in \mathcal{U}_{\mathrm{HL}}$ and, by definition of $\mathcal{U}$, we obtain $s \in \mathcal{U}$, as required. □

Finally, we have:

Theorem 7.4.11. *If* $s \in \mathcal{T}er(\Sigma_\bot, V)$, *then* $\mathrm{BLT}^{\infty}_{\mathrm{HL}}(s) = \mathrm{BLT}_{\mathrm{HL}}(s)$.

Proof. Suppose $\omega_{\mathcal{U}_{\mathrm{HL}}}$ is defined according to Definition 7.3.1, where $\overline{\mathcal{U}}_\bot$ is instantiated by $\mathcal{U}_{\mathrm{HL}}$. Moreover, suppose $s \in \mathcal{T}er(\Sigma_\bot, V)$. As $\omega_{\mathcal{U}_{\mathrm{HL}}}$ is defined by means of ω_{HL}, there exist $s \to^* t$ such that $\omega_{\mathcal{U}_{\mathrm{HL}}}(s) \preccurlyeq \omega_{\mathrm{HL}}(t)$. We next show $\omega_{\mathrm{HL}}(s) \preccurlyeq \omega_{\mathcal{U}_{\mathrm{HL}}}(s)$.

By definition of $\omega_{\mathcal{U}_{\mathrm{HL}}}$, we have that the subterms of s that are either in $\mathcal{U}_{\mathrm{HL}}$ or that are reducible to a redex of $\mathcal{R}$ are replaced by $\bot$. Hence, by Lemma 7.4.10 and the definition of direct approximants, it follows that all replaced subterms of s have $\bot$ as their Huet-Lévy direct approximant. By Lemma 5.4.9.(2) it now follows that $\omega_{\mathrm{HL}}(s) \preccurlyeq \omega_{\mathcal{U}_{\mathrm{HL}}}(s)$.

By the above facts relating ω_{HL} and $\omega_{\mathcal{U}_{\mathrm{HL}}}$, we have that $\mathrm{BLT}_{\mathcal{U}_{\mathrm{HL}}}(s) = \mathrm{BLT}_{\mathrm{HL}}(s)$. Hence, by Theorem 7.3.4, which we may apply by virtue of Lemma 7.4.9, we also have $\mathrm{BLT}^{\infty}_{\mathrm{HL}}(s) = \mathrm{BLT}_{\mathrm{HL}}(s)$. □

The above theorem yields an alternative proof showing that Huet-Lévy trees are congruent: Simply employ the theorem in conjunction with Theorem 7.2.12.

Discussion. The above definition of the set $\mathcal{U}_{\mathrm{HL}}$ actually differs slightly from the one given by Kennaway, Van Oostrom, and De Vries [KOV99]. Their definition requires one additional step and also an additional nullary function symbol $\mho$, where it is assumed that ω_{HL} is a map on $\mathcal{T}er(\Sigma_\mho, V)$ where $\mho$ takes on the rôle of $\bot$. The following sets are defined:

$$\mathcal{V}^{\mathrm{f}}_{\mathrm{HL}} = \{s \mid \forall s \to^* t : \omega_{\mathrm{HL}}(t) = \mho\}$$
$$\mathcal{V}_{\mathrm{HL}} = \{S \mid \forall S \twoheadrightarrow T : \forall t \in \mathcal{T}er(\Sigma_\bot, V) : t \preccurlyeq T \Longrightarrow t \in \mathcal{V}^{\mathrm{f}}_{\mathrm{HL}}\}$$

That is, the same sets as were defined at the beginning of this section, but with $\bot$ replaced by $\mho$. Next, $\mathcal{U}'_{\mathrm{HL}}$ is defined as the closure of $\mathcal{V}_\bot = \mathcal{V}_{\mathrm{HL}} \cup \{\bot\}$ under $\leftrightarrow_{\mathcal{V}_\bot}$. The set $\mathcal{U}'_{\mathrm{HL}}$ is shown to be a set of meaningless terms and it is employed to define Böhm-like trees.

Obviously, our definition of $\mathcal{U}_{\mathrm{HL}}$ omits from the definition of $\mathcal{U}'_{\mathrm{HL}}$ the function symbol $\mho$ and the step involving closure. Of course, we should ask ourselves if this is allowed. That is, two questions need to be answered: Is $\mathcal{U}_{\mathrm{HL}}$ really a set of meaningless terms and are the Böhm-like trees defined by $\mathcal{U}_{\mathrm{HL}}$ and $\mathcal{U}'_{\mathrm{HL}}$ identical?

The first question is implicitly answered by Kennaway, Van Oostrom, and De Vries. They show that $\mathcal{V}_{\mathrm{HL}}$ satisfies the last four clauses of Definition 7.2.1. Hence, $\mathcal{U}_{\mathrm{HL}}$ also satisfies the last four clauses. That the first clause is satisfied too, is immediate by the fact that $\bot \in \mathcal{U}^{\mathrm{f}}_{\mathrm{HL}}$ and Lemma 7.4.10.

That the defined Böhm-like trees are identical follows by the fact that the rewrite rule $\mho \to_\bot \bot$ occurs in the Böhm-like iTRS based on $\mathcal{U}'_{\mathrm{HL}}$ and by the

fact that $\mathcal{U}_{\mathrm{HL}}$ and $\mathcal{U}'_{\mathrm{HL}}$ are identical whenever $\bot$ and $\mho$ are identified, which is immediate by the construction of $\mathcal{U}'_{\mathrm{HL}}$ from $\mathcal{V}_{\mathrm{HL}}$ and the fact that $\mathcal{U}'_{\mathrm{HL}}$ is a set of meaningless terms.

7.4.4 Melting TRSs

In this section, we assume that we have at our disposal an orthogonal TRS $\mathcal{R} = (\Sigma, R)$ and an ω_{m}TRS $\mathcal{M} = (\Sigma_\bot, M)$ for $\mathcal{R}$ such that the rules of $\mathcal{M}$ and $\mathcal{R}$ do *not* overlap. Moreover, we assume that $\mathcal{L} = (\Sigma_\bot, L)$ is the ω_{r}TRS for $\mathcal{R}$ and that ω_{M} denotes the direct approximant function based on the ωTRS $\mathcal{LM} = (\Sigma_\bot, L \cup M)$.

Given the above assumptions, we show that it is possible to define for $\mathcal{M}$ a set $\mathcal{U}_{\mathrm{M}}$ of meaningless terms such that the Böhm-like trees based on $\mathcal{M}$ and $\mathcal{U}_{\mathrm{M}}$ are identical. To this end, we first define:

$$\mathcal{U}^{\mathrm{f}}_{\mathrm{M}} = \{s \mid \forall s \to^* t : \omega_{\mathrm{M}}(t) = \bot\}$$
$$\mathcal{U}_{\mathrm{M}} = \{S \mid \forall S \twoheadrightarrow T : \forall t \in \mathcal{T}er(\Sigma_\bot, V) : t \preccurlyeq T \Rightarrow t \in \mathcal{U}^{\mathrm{f}}_{\mathrm{M}}\}$$

We have the following:

Lemma 7.4.12. *The set $\mathcal{U}_{\mathrm{M}}$ is a set of meaningless terms.*

Proof. The only non-trivial clauses are the third and fifth clause. The third clause follows by the non-overlap between $\mathcal{M}$ and $\mathcal{R}$. The fifth clause follows by a similar argument as the one given in Section 8.1.4 of [KOV99] for $\mathcal{U}_{\mathrm{HL}}$. □

Denoting the Böhm-like tree based on $\mathcal{U}_{\mathrm{M}}$ by $\mathrm{BLT}^{\infty}_{\mathrm{M}}$ and the Böhm-like tree based on $\mathcal{M}$ by $\mathrm{BLT}_{\mathrm{M}}$, we can now prove:

Theorem 7.4.13. *If $s \in \mathcal{T}er(\Sigma_\bot, V)$, then $\mathrm{BLT}^{\infty}_{\mathrm{M}}(s) = \mathrm{BLT}_{\mathrm{M}}(s)$.*

Proof. The proof from the previous section showing that $\mathrm{BLT}^{\infty}_{\mathrm{HL}}$ and $\mathrm{BLT}_{\mathrm{HL}}$ are identical and the lemmas employed therein only use the fact that ω_{HL} can be defined by means of an ωTRS. Hence, as ω_{M} is also defined by means of an ωTRS, we have that the proof from the previous section carries over immediately. □

Open Problem. Given the assumptions at the beginning of the current section, one question remains: Is it really required to assume there is no overlap between $\mathcal{M}$ and $\mathcal{R}$? To show that it is not required, it has to be proved that Lemma 7.4.12 holds under assumption of overlap. The problematic part herein is showing that the third clause of Definition 7.2.1 is satisfied, since overlap is involved in the clause.

7.5 Summary

The table depicted below summarises the relation between Böhm-like trees respectively defined by sets of meaningless terms (denoted by $\mathcal{U}$), direct approximant functions (denoted by ω), ωTRSs, and ω_{m}TRSs.

The first entry of each row defines which Böhm-like tree we start out with. The other entries of each row specify whether an identical Böhm-like tree can be defined of the type that heads the particular column in which the entry occurs. Here, a checkmark (✓) indicates that an identical tree always exists and a dash (-) indicates that this may not be the case, while a question mark (?) indicates that it is unknown. The numbers correspond to those below the table.

	To			
From	$\mathcal{U}$	ω	ωTRS	ω_{m}TRS
$\mathcal{U}$	✓(1)	✓(2)	– (3)	– (3)
ω	– (4)	✓(1)	– (5)	– (5)
ωTRS	– (4)	✓(6)	✓(1)	– (7)
ω_{m}TRS	? (8)	✓(9)	✓(9)	✓(1)

1. This holds trivially, since it concerns the same definition twice.
2. This holds by Theorem 7.3.4.
3. This does not hold, as Berarducci-like trees can be defined by a set of meaningless terms but not by ωTRSs and ω_{m}TRSs (see Chapters 5 and 6).
4. This does not hold, as Böhm-like trees based on direct approximant functions and ωTRSs are not necessarily congruent (see Chapter 6), while congruence always holds for Böhm-like trees based on sets of meaningless terms by Theorem 7.2.12.
5. This does not hold, as Berarducci-like trees can be defined by means of direct approximant functions but not by means of ωTRSs or ω_{m}TRSs (see Chapters 5 and 6).
6. This holds by Theorem 5.4.10.
7. This does not hold, as Böhm-like trees based on ωTRSs are not necessarily congruent, while congruence always holds for Böhm-like trees based on ω_{m}TRSs (see Chapter 6).
8. This holds by Theorem 7.4.13 in case there is no overlap between $\mathcal{R}$ and the ω_{m}TRS. Otherwise, it is an open problem.
9. This holds by Proposition 6.4.7 and Theorem 5.4.10.

Hence, most flexibility is provided by direct approximant functions. While the other three approaches, i.e., sets of meaningless terms, ωTRSs, ω_{m}TRSs, have their limitations.

8

Sequentiality

"It's Trillian!" shouted Arthur. "Or is it... er... God, I can't stand all this parallel universe stuff."

— Douglas Adams
Mostly Harmless (1992)

In this chapter, we employ the Böhm-like trees developed in Chapters 5 and 6 to show that certain *sequential* orthogonal constructor TRSs are indeed *sequential*. Although proving such a statement seems trivial, it is not: There are two kinds of sequentiality in play here.

Parallelism. To understand the two kinds of sequentiality, it is probably easiest to first consider the different ways in which the word *parallel* is utilised throughout computer science. Essentially, there are two such ways:

1. parallel is short for *parallel execution*, and
2. parallel is synonymous to *concurrent*.

We discuss each of the above two utilisations in turn.

To be able to *execute in parallel* some parts of a program signifies that the parts may be executed at the *same* time either on different processors or on different computers. Prerequisite for parallel execution is the requirement that the parts do not need each other's outputs. If one particular part does need the output of some other part, then the particular part has to wait until the output is available. This implies that the parts can *only* be executed in succession and not at the same time.

Both TRSs and the $\lambda\beta$-calculus allow for parallel execution in the form of parallel reduction: Subterms that occur at parallel positions may be reduced (executed) at the same time. The reduction of a subterm only depends on the structure of that subterm and not on what occurs at either prefix positions or parallel positions.

To be able to write a program that is *concurrent* signifies that it is possible to write code that calls multiple functions in such a way that the code can produce output if it obtains the output of a subset of the functions and if there is no function whose output is always required.

The prime example of a concurrent program is the *parallel-or*. Employing a functional programming style, the parallel-or is defined as follows:

$$\begin{aligned} \mathit{ParallelOr}(\mathit{True}, x) &\to \mathit{True} \\ \mathit{ParallelOr}(x, \mathit{True}) &\to \mathit{True} \\ \mathit{ParallelOr}(\mathit{False}, \mathit{False}) &\to \mathit{False} \end{aligned}$$

Since argument evaluation in functional languages essentially consists of function calls, *ParallelOr* is truly a concurrent program: If one of the arguments evaluates

to *True*, then evaluating the other argument is unnecessary. That is, there exists a subset of functions and there is no function whose output is always required.

Constructs that facilitate concurrency occur in many different guises in programming languages. This varies from add-on libraries, like threading and message passing libraries, to specialised language constructs, like the `in`, `read`, and `out` constructs of Linda, a programming language developed by Carriero, Gelernter, and co-workers [ACG86]. The constructs are used in Linda to address TupleSpaces: `in` adds a tuple to a tuple space, `read` reads a tuple, and `out` reads and removes a tuple. Concurrency is achieved by injecting the results of the called functions into a tuple space and letting the caller remove tuples based on a selection of the components of the tuples, which may be *equal* for a number of the called functions.

It is important to note that parallel execution and concurrency are independent concepts: Parallel execution may be possible while concurrency is not, and vice versa. Of course, parallel execution and concurrency often occur together.

An example of a system in which parallel execution is possible, but in which concurrency is not, is the $\lambda\beta$-calculus: Parallel execution is possible in the form of parallel reductions. Concurrency is not possible as witnessed by the sequentiality property mentioned in Chapter 4 (see also below).

A system in which concurrency is possible, but in which parallel execution is not, is any run-of-the-mill (single-core) single processor computer running an operating system like Linux or Windows: Concurrency is possible by virtue of preemptive multitasking (essentially interleaving). Parallel execution is not possible, as there is only one (single-core) processor (see Tanenbaum [Tan01]).

Sequentiality. Given the two utilisations of the word parallel, it is now easy to explain the two kinds of sequentiality: The two kinds are just the negations of the two utilisations of the word parallel, i.e., *not* executed in parallel and *not* concurrent. Not being concurrent means there is a fixed function that always has to produce output.

Not executing in parallel and still obtaining the same output as in a parallel execution is always possible: Simply execute those parts that need to be executed in parallel in an *interleaved* fashion. Of course, interleaved execution (and also parallel execution) may be quite wasteful, as part of what is executed may not contribute to the output. Research has been done on sequential executions where only those parts are executed that are *really needed* to obtain output (see below). As might be expected, the existence of such a strategy for a specific program is undecidable in general.

Implementing a concurrent program in a programming language that does not allow for concurrency requires a serious effort from the programmer: It is impossible to choose a subset of required functions in advance, since it is in general undecidable which functions actually produce output. Hence, the programmer needs to implement some interleaving scheme that alternates between the executions of the different functions. Such an alternation scheme generally requires some form of *continuations* (see, e.g., the book by De Bakker and De Vink [BV96]).

Term Rewriting. To define both kinds of sequentiality in the context of term rewriting, it is first necessary to define what 'to produce output' means. In the

literature, two definitions are common: To produce output either means to reach a *normal form* or it means to reach a *root-stable term*, which is also called a *head normal form* in the current context.

Taking the stance that to produce output means to reach a head normal form, and supposing orthogonal TRSs, we next describe informally both types of sequentiality in the context of rewriting. A completely analogous description can be given in case the stance is taken that to produce output means to reach a normal form.

Not reducing (executing) in parallel obviously means that only one redex is contracted at a time. Finding a head normal form, if one exists, by contracting only one redex at a time is always possible: outermost fair reductions are head normalising. Since sequential reductions essentially interleave parallel reductions, one might suspect there exists a parallel analogue to the theorem regarding outermost fair reductions. This is actually the case: parallel outermost reductions are also head normalising.

To understand non-concurrency in the context of rewriting, suppose we have at our disposal a non-root-stable term s of the form $f(s_1, \ldots, s_n)$ and a number of rewrite rules whose left-hand sides have the form $f(t_1, \ldots, t_n)$ for different t_i. Given that the reduction of the arguments of f in s is the term rewriting equivalent of a number of function calls, non-concurrency implies that there are fixed arguments s_i of s that must be reduced to head normal form before a redex is created at the root. Concurrency implies that there are no fixed arguments s_i.

Even in case of orthogonality there does not need to be a fixed argument. This is exemplified by *Gustave's function*, as first defined by Berry [Ber76]:

$$f(a, b, x) \to c$$
$$f(x, a, b) \to c$$
$$f(b, x, a) \to c$$

Gustave's function is orthogonal due to the different ways in which the nullary function symbols a and b occur as arguments of f. Moreover, there is no fixed argument that always needs to be evaluated due to the variable x which occurs at three different argument positions in the three rewrite rules.

Below, we employ Böhm-like trees to formally define non-concurrency. Böhm-like trees provide a suitable formalism by the fact that they, just like concurrency and non-concurrency, involve the concept of root-stability. For this reason, we call sequentiality in the sense of non-concurrency *Böhm-like tree sequentiality*.

Defining non-concurrency by means of Böhm-like trees presupposes that the employed Böhm-like trees actually represent a 'sufficient' part of the produced output. Trivial trees, e.g., will not do. As shown in Section 8.3, these trees always satisfy non-concurrency, independent of any actual concurrency present in the assumed TRS. Whence, trivial trees can, e.g., be used to show that the parallel-or is non-concurrent, which is obviously an invalid statement.

Needed Redexes. Although the above describes both non-parallel reductions and non-concurrency in the context of term rewriting, it does not describe sequential reductions (executions) that only reduce those parts needed to obtain output. Describing such reductions is facilitated by the concept of a *(head) needed redex*. That

is, a redex a residual of which is contracted in any reduction to (head) normal form. Here, either a normal form or a head normal form is employed depending on the definition of producing output. Obviously, since it is in general undecidable if a term is reducible to some other term, neededness is undecidable too.

Given the concept of a needed redex, sequential reductions that only reduce those parts needed to obtain output can now be defined as non-parallel reductions contracting only needed redexes. Since neededness is undecidable, it is also undecidable if non-parallel reductions exist that only reduce needed redexes. However, it is possible to define certain classes of TRSs for which neededness, and, hence, the existence of non-parallel reductions contracting only needed redexes, is decidable. An overview of the most well-known classes is given by Durand and Middeldorp [DM97].

One of the classes of TRSs for which neededness is decidable was first defined by Huet and Lévy [HL91]: the class of *strongly sequential orthogonal TRSs*. Most confusingly, the class is defined by means of concepts that derive from the definition of non-concurrency. However, this also leads to the hypothesis that strongly sequential orthogonal TRSs are non-concurrent and this is what we actually prove in this chapter in the case of strongly sequential orthogonal *constructor* TRSs.

Remark 8.0.1. Numerous publications concerning rewriting, like those by Berry [Ber78a, Ber78b] and by Huet and Lévy [HL91], confuse sequentiality in the sense of non-concurrency with sequential reductions in which only needed redexes are contracted. The confusion comes about by the supposition that sequentiality in the sense of non-concurrency is required to obtain sequential reductions in which only needed redexes are contracted. However, as examples by Durand and Middeldorp [DM97, Lemma 31] show, the supposition is incorrect.

Overview. The remainder of this chapter is structured as follows: In Section 8.1, we define both Böhm-like tree sequentiality (non-concurrency) and strong sequentiality. Thereafter, in Section 8.2, Böhm-like tree sequentiality is shown to hold for strongly sequential orthogonal constructor TRSs under assumption of a Böhm-like tree of which it is reasonable to say that it represents a 'sufficient' part of the produced output. In Section 8.3, Böhm-like tree sequentiality is shown for two Böhm-like trees of which it *cannot* be reasonably said that they represent a 'sufficient' part of the produced output. In Section 8.4 some open problems regarding sequentiality are presented. Finally, in Section 8.5, it is shown that stability always holds under assumption of Böhm-like tree sequentiality. Hence, stability is a slightly weaker property than sequentiality.

8.1 Definitions

We define Böhm-like tree sequentiality (non-concurrency) and strong sequentiality. Böhm-like tree sequentiality is defined in Section 8.1.1. Strong sequentiality is defined in Section 8.1.2 together the concept of *normal form sequentiality*. In Section 8.1.3, a comparison is made between Böhm-like tree sequentiality and normal form sequentiality.

8.1.1 Böhm-Like Tree Sequentiality

Assuming that $\mathcal{R} = (\Sigma, R)$ is a left-linear TRS and that BLT is a *monotone* Böhm-like tree for $\mathcal{R}$, we define Böhm-like tree sequentiality:

Definition 8.1.1. *The TRS $\mathcal{R}$ is* Böhm-like tree sequential *with respect to* BLT*, if for every $s \in \mathcal{T}er(\Sigma_\perp, V)$ and every $p \in \mathcal{P}os(\mathrm{BLT}(s))$ with $\mathrm{BLT}(s)|_p = \{\perp\}$ exactly one of the following holds:*

1. *for all $t \in \mathcal{T}er(\Sigma_\perp, V)$, if $t \not\succcurlyeq s$, then $\mathrm{BLT}(t)|_p = \{\perp\}$, or*
2. *there exist $q \in \mathcal{P}os(s)$ with $s|_q = \perp$ such that for all $t \in \mathcal{T}er(\Sigma_\perp, V)$, if $t \not\succcurlyeq s$ and $\mathrm{BLT}(t)|_p \neq \{\perp\}$, then $t|_q \neq \perp$.*

Remark that we can assume that $p \in \mathcal{P}os(\mathrm{BLT}(t))$, since BLT is assumed to be monotone. By monotonicity, $s \preccurlyeq t$ implies $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$, from which it follows immediately that $\mathcal{P}os(\mathrm{BLT}(s)) \subseteq \mathcal{P}os(\mathrm{BLT}(t))$ and $p \in \mathcal{P}os(\mathrm{BLT}(t))$.

As explained in the introduction of this chapter, the Böhm-like trees in the above definition serve the purpose of representing the produced output, where the produced output is defined by root-stability. The actual encoding of non-concurrency is captured by the second clause of the definition: Employing $\perp$ as a placeholder for the arguments, the second clause points out the arguments, the positions q, that always need to be evaluated to produce output. The first clause is there to deal with the situation in which a subterm never becomes root-stable.

Contrary to what is sometimes thought, syntactic continuity of Böhm-like trees (see Chapter 6) is not a prerequisite for Böhm-like tree sequentiality. To see this, consider the TRS presented in Section 6.1:

$$\begin{aligned} \mathit{IsEmpty}(\mathit{nil}) &\to \mathit{True} \\ \mathit{IsEmpty}(x : xs) &\to \mathit{False} \end{aligned}$$

As explained in Section 6.1, the following rules define an ωTRS:

$$\begin{aligned} \mathit{IsEmpty}(xs) &\to_\omega \perp \\ \mathit{nil} &\to_\omega \perp \end{aligned}$$

As also explained, the ωTRS does not define a Böhm-like tree that satisfies syntactic continuity. However, the tree does satisfy Böhm-like tree sequentiality, by the simple fact that *IsEmpty* has only one argument and the fact that Böhm-like trees defined by ωTRSs are always monotone.

Bibliographic Notes. The definition of Böhm-like tree sequentiality finds its origins in the work on *concrete domains* by Kahn and Plotkin [KP93]. The first formulation in the context of Böhm-like trees is by Berry [Ber78a], who proves that the $\lambda\beta$-calculus is Böhm-like tree sequential under assumption of Böhm trees. More precisely, Berry proves a slightly stronger property: There is a unique position q in the second clause of Definition 8.1.1 in his case (see Section 4.3.2). Kahn and Plotkin do not require uniqueness in the second clause.

More details regarding Böhm-like tree sequentiality can be found in Ong's handbook chapter [Ong95]. As no other forms of sequentiality occur in his chapter, Ong calls Böhm-like tree sequentiality simply sequentiality.

8.1.2 Strong Sequentiality

We next define strong sequentiality and recall some properties related to strong sequentiality. To be able to define strong sequentiality, we first define sequential predicates and normal form sequentiality. Both definitions assume that $\mathcal{R} = (\Sigma, R)$ is a *confluent, left-linear* TRS.

Sequential predicates require us to define sequentiality indices:

Definition 8.1.2. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$ and $s|_p = \perp$ for some $p \in \mathcal{P}os(s)$. If P is a monotone predicate on $\mathcal{T}er(\Sigma_\perp, V)$, then the position p is a* sequentiality index *of s when for all $t \succcurlyeq s$ with $P(t) = \mathsf{T}$ it holds that $t|_p \neq \perp$.*

Above, we call a predicate monotone if it is monotone as a map from $\mathcal{PO} = (\mathcal{T}er(\Sigma_\perp, V), \preccurlyeq)$ to $\mathcal{TT} = (\{\mathsf{T}, \mathsf{F}\}, \sqsubseteq)$, where $\mathsf{F} \sqsubseteq \mathsf{T}$ and $\mathsf{T} \not\sqsubseteq \mathsf{F}$. Moreover, given a monotone predicate P and a term $s \in \mathcal{T}er(\Sigma_\perp, V)$ such that $P(s) = \mathsf{F}$, we denote by $\mathcal{I}_P(s)$ the set of all sequentiality indexes of s.

Now that we know what a sequentiality index is, we can define:

Definition 8.1.3. *Let P be a monotone predicate on $\mathcal{T}er(\Sigma_\perp, V)$ and let $s \in \mathcal{T}er(\Sigma_\perp, V)$. The predicate P is* sequential *in s if $P(s) = \mathsf{F}$ and if the existence of a term $t \succcurlyeq s$ with $P(t) = \mathsf{T}$ implies $\mathcal{I}_P(s) \neq \emptyset$.*

With respect to $\mathcal{R}$, we can define the following predicate on *partial* terms:

$$\mathsf{NF}(s) = \begin{cases} \mathsf{T} & \text{if } s \text{ has a normal form in } \mathcal{T}er(\Sigma, V) \\ \mathsf{F} & \text{if } s \text{ does not have a normal form in } \mathcal{T}er(\Sigma, V) \end{cases}$$

Observe that $\mathcal{T}er(\Sigma, V)$ is employed in the above definition and *not* $\mathcal{T}er(\Sigma_\perp, V)$. We have by left-linearity of $\mathcal{R}$ that $s \preccurlyeq t$ implies $\mathsf{NF}(s) \sqsubseteq \mathsf{NF}(t)$, i.e., NF is a monotone predicate.

Given the predicate NF, we can define:

Definition 8.1.4. *The TRS $\mathcal{R}$ is* normal form sequential *if and only if NF is a sequential predicate in every element of $\mathcal{T}er(\Sigma_\perp, V)$.*

An example of an *orthogonal* TRS which is not normal form sequential is Gustave's function from the introduction. To see that the function is not normal form sequential, consider the term $f(\perp, \perp, \perp)$, which does not have a normal form in $\mathcal{T}er(\Sigma, V)$. Obviously, we have that $f(a, b, \perp)$, $f(\perp, a, b)$, and $f(b, \perp, a)$ all have a normal form in $\mathcal{T}er(\Sigma, V)$ and that all have $f(\perp, \perp, \perp)$ as prefix. However, since there is no argument unequal to $\perp$ that is shared between the terms $f(a, b, \perp)$, $f(\perp, a, b)$, and $f(b, \perp, a)$, Gustave's function is not normal form sequential.

We have the following lemma:

Lemma 8.1.5. *If $\mathcal{R}$ is orthogonal, then it is normal form sequential if and only if NF is a sequential predicate in every normal form of $\mathcal{T}er(\Sigma_\perp, V)$.*

Proof. This is Lemma 4.6 of [HL91]. □

Given the definition of normal form sequentiality, we are now almost in a position to define strong sequentiality. The only missing piece of the puzzle is an

approximation of the TRS $\mathcal{R}$. The approximation, denoted $\mathcal{R}_? = (\Sigma, R_?)$, has the following set of rewrite rules:

$$R_? = \{l \to \rho(l) \mid l \to r \in R\}\,,$$

where ρ is any map from $\mathcal{T}er(\Sigma_\perp, V)$ to V that satisfies $\rho(l) \notin \mathcal{V}ar(l)$. Since the left-hand sides of $\mathcal{R}_?$ are precisely the left-hand sides of $\mathcal{R}$, it is immediate that $\mathsf{NF}_\mathcal{R} = \mathsf{NF}_{\mathcal{R}_?}$. That $\to^*_\mathcal{R} \subseteq \to^*_{\mathcal{R}_?}$ is immediate by $\to_\mathcal{R} \subseteq \to_{\mathcal{R}_?}$, which follows by the fact that ρ satisfies $\rho(l) \notin \mathcal{V}ar(l)$, implying that any term and, hence, also the right-hand side of any rewrite rule of $\mathcal{R}$ may be substituted for $\rho(l)$.

Example 8.1.6. Supposing that $\mathcal{R}$ is actually Combinatory Logic, we have that $R_?$ consists of the following three rewrite rules:

$$\begin{aligned} Sxyz &\to v \\ Kxy &\to v \\ Ix &\to v \end{aligned}$$

where v is a variable different from x, y, and z.

Remark that *any* term s has a normal form with respect to $\mathcal{R}_?$: Rewriting any redex in s to a fresh variable, which is possible by definition of $\mathcal{R}_?$, reduces the number of functions symbols in s. Hence, as the number of functions symbols in s is finite, eventually a normal form is reached.

Given $\mathcal{R}_?$, we can now define:

Definition 8.1.7. *Let $\mathcal{R}$ be orthogonal. The TRS $\mathcal{R}$ is* strongly sequential *if and only if $\mathcal{R}_?$ is normal form sequential in every normal form of $\mathcal{T}er(\Sigma_\perp, V)$.*

As shown by Huet and Lévy [HL91, Section 4.2], any strongly sequential system is also normal form sequential. For ample motivation of the definition of strong sequentiality and examples of strongly sequential systems, the reader is referred to the papers by Huet and Lévy [HL91] and by Klop and Middeldorp [KM91].

Given a term s, we denote by $\mathcal{I}(s)$ the set $\mathcal{I}_\mathsf{NF}(s)$, where NF is interpreted with respect to $\mathcal{R}_?$. We next recall a number of properties of $\mathcal{I}(s)$ that are known from the literature.

The following two propositions hold:

Proposition 8.1.8. *Let $\mathcal{R}$ be orthogonal and let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$. If $s \preccurlyeq t$, $p \in \mathcal{I}(s)$, and $q \leqslant p$, then $p \in \mathcal{I}(t[\perp]_p)$ and $q \in \mathcal{I}(s[\perp]_q)$.*

Proof. This is Proposition 4.1 of [KM91]. □

Proposition 8.1.9. *Let $\mathcal{R} = (\Sigma, R)$ be orthogonal and let $s \preccurlyeq l$ for some $l \to r \in R$. If $s|_p = \perp$ and $l|_p \in V$ for some $p \in \mathcal{P}os(s)$, then $p \notin \mathcal{I}(s)$.*

Proof. Suppose $s|_p = \perp$ and $l|_p \in V$ for some $p \in \mathcal{P}os(s)$. Define $t = s \sqcap l$. By definition of t, we have $t|_p = \perp$ and $t = \sigma(l)$ for some substitution σ. As $t = \sigma(l) \to x$ for some $x \in V$, we have $\mathsf{NF}(t) = \mathsf{T}$ with respect to $\mathcal{R}_?$. Hence, as we also have $s \preccurlyeq t$ and $t|_p = \perp$, it holds that $p \notin \mathcal{I}(s)$. □

In case $\mathcal{R}$ is an orthogonal *constructor* TRS, we have:

Proposition 8.1.10. *Let $\mathcal{R}$ be an orthogonal constructor TRS. The TRS $\mathcal{R}$ is strongly sequential if and only if for all $s \preccurlyeq l$ with $l \to r \in R$ and s not an $l \to r$-redex it holds that $\mathcal{I}(s) \neq \emptyset$.*

Proof. This is Corollary 7.7 of [KM91]. □

Thus, for orthogonal constructor TRSs we have that they are sequential if and only if we have $\mathcal{I}(s) \neq \emptyset$ for every term s which is a prefix of a left-hand side of a rewrite rule, but which is *not* a substitution instance of the rewrite rule. Remark that for each such s there is at least one $p \in \mathcal{P}os(s)$ such that $s|_p = \bot$.

Bibliographic Notes. Both the concepts of normal form sequentiality and strong sequentiality originate in the work of Huet and Lévy [HL91]. In their work, normal form sequentiality is simply called sequentiality.

Like Böhm-like tree sequentiality, both normal form sequentiality and strong sequentiality derive from sequentiality as first defined by Kahn and Plotkin [KP93]. This is easily seen when realising that Böhm-like tree sequentiality is actually an instance of sequentiality as presented in Definition 8.1.3. In the case of Böhm-like tree sequentiality, the value of $P(s)$ is defined as $\mathrm{BLT}(s)|_p = \bot$.

Other important references on normal form sequentiality and strong sequentiality are the papers by Klop and Middeldorp [KM91] and by Comon [Com00].

8.1.3 Comparison

Since normal forms are special instances of head normal forms (normal forms are root-stable), it might be thought that each left-linear TRS that is Böhm-like tree sequential is also normal form sequential. It turns out that this is actually the case given one side-condition, as we show next.

Assuming that $\mathcal{R}$ is a confluent, left-linear TRS which is Böhm-like tree sequential with respect to some monotone Böhm-like tree BLT, we have the following:

Proposition 8.1.11. *If for each normal form $n \in \mathcal{T}er(\Sigma, V) \subseteq \mathcal{T}er(\Sigma_\bot, V)$ it holds that $\mathrm{BLT}(n) = {\downarrow}\{n\}$, then $\mathcal{R}$ is normal form sequential.*

Proof. Suppose for every normal form $n \in \mathcal{T}er(\Sigma, V)$ that $\mathrm{BLT}(n) = {\downarrow}\{n\}$. Let $s \in \mathcal{T}er(\Sigma_\bot, V)$ such that $\mathsf{NF}(s) = \mathsf{F}$. Remark that finiteness of $\mathrm{BLT}(s)$ implies there exists a position $p \in \mathcal{P}os(\mathrm{BLT}(s))$ such that $\mathrm{BLT}(s)|_p = \{\bot\}$. Otherwise, $\mathrm{BLT}(s) = {\downarrow}\{n\}$ for some normal form $n \in \mathcal{T}er(\Sigma, V)$, which implies $\mathsf{NF}(s) = \mathsf{T}$.

If there exist $t \succcurlyeq s$ such that $\mathsf{NF}(t) = \mathsf{T}$, then, by definition of NF, there exists a normal form $n \in \mathcal{T}er(\Sigma, V)$ such that $t \to^* n$ and $\mathrm{BLT}(t) = {\downarrow}\{n\}$. By monotonicity of BLT this implies that $\mathrm{BLT}(s)$ is finite. Hence, there exist $p \in \mathcal{P}os(\mathrm{BLT}(s))$ such that $\mathrm{BLT}(s)|_p = \{\bot\}$ and $\mathrm{BLT}(t)|_p \neq \{\bot\}$.

Since the above holds for all $t \succcurlyeq s$ with $\mathsf{NF}(t) = \mathsf{T}$, where the positions $p \in \mathcal{P}os(\mathrm{BLT}(s))$ are obviously the same, there exists by Böhm-like tree sequentiality a position $q \in \mathcal{P}os(s)$ such that $s|_q = \bot$ and $t|_q \neq \bot$. Hence, we have $q \in \mathcal{I}_{\mathsf{NF}}(s)$, as required for $\mathcal{R}$ to be normal form sequential. □

To see that the condition regarding normal forms cannot be omitted from the above proposition, even in case $\mathcal{R}$ is orthogonal, consider again Gustave's function from the introduction of this chapter and define the following map from partial terms to partial terms:

$$\omega(s) = \begin{cases} s & s \text{ normal form and } s \in \mathcal{T}er(\{a, b, f\}, V) \\ \bot & \text{otherwise} \end{cases}$$

The map ω is a monotone direct approximant function, as is readily shown. Hence, ω defines a monotone and continuous Böhm-like tree, which we denote BLT. However, $\mathrm{BLT}(c) = \{\bot\}$, which violates the side condition of the lemma.

The Böhm-like tree also ensures that $\mathcal{R}$ is Böhm-like tree sequential. This is easy to show except for $f(\bot, \bot, \bot)$ occurs. In the case of $f(\bot, \bot, \bot)$, a redex can be created by replacing two of the arguments of f by a and b respectively. However, there is no argument that always needs to be replaced to create a redex. Despite this, Böhm-like tree sequentiality holds, since each possible redex that can be created reduces the term to c which yields $\omega(f(\bot, \bot, \bot)) = \omega(c) = \bot$.

Remark that Gustave's function is not normal form sequential. Hence, in case there is a normal form $n \in \mathcal{T}er(\Sigma, V)$ for which we do not have $\mathrm{BLT}(n) = {\downarrow}\{n\}$, we can have Böhm-like tree sequentiality while we do not have normal form sequentiality. Of course, since $\mathrm{BLT}(c) \neq {\downarrow}\{c\}$, it is reasonable to argue that the part of the output represented by BLT is *not* sufficient.

8.2 Sequentiality

In this section, we show that Böhm-like tree sequentiality holds for strongly sequential orthogonal constructor TRSs under assumption of a Böhm-like tree of which it is reasonable to say that it represents a 'sufficient' part of the produced output. The Böhm-like tree is defined in Section 8.2.1. Thereafter, in Section 8.2.2, a structure is defined that facilitates the proof of Böhm-like tree sequentiality. In Section 8.2.3, the actual proof of Böhm-like tree sequentiality occurs.

8.2.1 Böhm-Like Tree

We next define the Böhm-like tree that we employ to prove that strongly sequential orthogonal constructor TRSs are Böhm-like tree sequential.

As explained in the introduction of this chapter, we require a Böhm-like tree that represents a 'sufficient' part of the produced output. Ideally, the sufficient part includes all the produced output, i.e., everything that becomes root-stable in maximal fair reductions. Defining sufficient in this way limits the choice of the Böhm-like tree to the Berarducci-like tree. Unfortunately, the Berarducci-like tree is not monotone (see Chapter 5). Hence, as monotonicity is a prerequisite for Böhm-like tree sequentiality, the Berarducci-like tree cannot be employed.

By the above, we can only include produced output as long as monotonicity is preserved. Hence, given a term s, we could define sufficient as everything that

becomes root-stable in the maximal fair reductions of all $t \succcurlyeq s$. However, since we are considering constructor TRSs and since it may be argued that only the constructors of such TRSs count as output, it seems more reasonable to only require that the constructors that do not occur in subterms with a defined symbol at the root are included in the definition of sufficient; all else included can be seen as added bonus.

Summarising the above, we require the Böhm-like tree we employ below to satisfy the following two requirements: First, it must be monotone. Second, for each term s, if a constructor occurs at position p in s, then the constructor must also occur at position p in the Böhm-like tree of s as long as no defined symbols occur at prefix positions of p in s. Remark that each such constructor is root-stable, because each rewrite rule has a defined symbol at its root.

We have already encountered a Böhm-like tree that satisfies the above two requirements: the Huet-Lévy tree. As explained in Chapter 5, the tree is monotone. Moreover, the tree satisfies the second requirement, as the left-hand side of each rewrite rule of the Huet-Lévy ωTRS will have a defined symbol at the root in case a constructor TRS is assumed.

Unfortunately, we cannot prove Böhm-like tree sequentiality for strongly sequential orthogonal constructor TRSs under the assumption of Huet-Lévy trees. To see that such a proof is impossible, consider the orthogonal constructor TRS which consists only of the following rewrite rule:

$$f(c, c) \to c\,.$$

To show that the TRS satisfies strong sequentiality Proposition 8.1.10 requires us to prove that $\mathcal{I}(s)$ is non-empty in case s is equal to either $f(\bot, \bot)$, $f(\bot, c)$, $f(c, \bot)$, or $\bot$. This is immediate, since no variables occur in $f(c, c)$.

The rewrite rules of the Huet-Lévy ωTRS are as follows:

$$\begin{aligned} f(c, c) &\to_\omega \bot & \qquad f(c, \bot) &\to_\omega \bot \\ f(\bot, c) &\to_\omega \bot & \qquad f(\bot, \bot) &\to_\omega \bot \end{aligned}$$

Hence, we have for $s = f(\bot, \bot)$ that $\mathrm{BLT}_{\mathrm{HL}}(s) = \{\bot\}$ and $\mathrm{BLT}_{\mathrm{HL}}(s)|_\epsilon = \{\bot\}$.

Considering the terms $f(\bot, x) \succcurlyeq s$ and $f(x, \bot) \succcurlyeq s$, we obtain:

$$\begin{aligned} \mathrm{BLT}_{\mathrm{HL}}(f(\bot, x)) &= {\downarrow}\{f(\bot, x)\} \neq \{\bot\} \\ \mathrm{BLT}_{\mathrm{HL}}(f(x, \bot)) &= {\downarrow}\{f(x, \bot)\} \neq \{\bot\} \end{aligned}$$

Hence, the first clause of Definition 8.1.1 is not satisfied. The second clause of Definition 8.1.1 is not satisfied either, since we have:

$$\begin{aligned} f(\bot, x)|_1 &= \bot \neq x = f(x, \bot)|_1 \\ f(\bot, x)|_2 &= x \neq \bot = f(x, \bot)|_2 \end{aligned}$$

Hence, the assumed strongly sequential orthogonal construct TRS is not Böhm-like tree sequential with respect to the Huet-Lévy tree.

Böhm-like tree sequentiality does not hold in the above example by the presence of the rule $f(\bot, \bot) \to_\omega \bot$ in the Huet-Lévy ωTRS. The rule applies to the term $f(\bot, \bot)$, but not to the terms $f(x, \bot)$ and $f(\bot, x)$. To overcome this problem, we could replace the rule $f(\bot, \bot) \to_\omega \bot$ by the following two rewrite rules:

$$f(\bot, x) \to_\omega \bot$$
$$f(x, \bot) \to_\omega \bot$$

The above two rewrite rules together with the three rewrite rules of the Huet-Lévy ωTRS that were not replaced, are readily shown to define an ωTRS.

Denoting the Böhm-like tree based on the ωTRS by BLT, we obtain:

$$\mathrm{BLT}(f(\bot, \bot)) = \mathrm{BLT}(f(\bot, x)) = \mathrm{BLT}(f(x, \bot)) = \{\bot\}\,.$$

Hence, Böhm-like tree sequentiality is satisfied by the *terms* considered above. In addition, the Böhm-like tree still satisfies the proposed definition of a sufficient part of the produced output, as all rewrite rules of the ωTRS have a defined symbol at the root.

Employing the above scheme of replacing a left-hand side in which $\bot$ occurs multiple times by a number of left-hand sides in which each $\bot$, except for *one*, is replaced by a unique fresh variable, we define a new ωTRS. As we will see, the ωTRS satisfies the proposed definition of a sufficient part of the produced output. Moreover, as we show in Section 8.2.3, the Böhm-like tree based on the ωTRS allows us to prove that strongly sequential orthogonal constructor TRSs are Böhm-like tree sequential.

In the definition given below, we do not introduce all possible left-hand sides, given the above replacement scheme. The introduction of a certain left-hand side depends on the sequentiality indices of the left-hand side in which $\bot$ occurs multiple times. Assuming in the remainder of this section that $\mathcal{R} = (\Sigma, R)$ is a strongly sequential orthogonal constructor TRS, we define the new ωTRS by means of an ω_{m}TRS (see Chapter 6):

Definition 8.2.1. *The ω_{m}TRS $\mathcal{M}$ is defined as $\mathcal{M} = (\Sigma_\bot, M)$, where:*

$$M = \{s \to_{\mathrm{m}} \bot \mid s \in \mathit{pattern}_\mathcal{R}\}\,,$$

with $\mathit{pattern}_\mathcal{R}$ as defined in Figure 8.1.

In Figure 8.1, $\mathcal{D}$ denotes the set of defined symbols and $v(s)$ denotes the term s with *each* $\bot$ replaced by a unique fresh variable. Moreover, $\mathit{pattern}^*_\mathcal{R}$ facilitates the introduction of left-hand sides for all prefixes of left-hand sides of $\mathcal{R}$, as far as these can be deemed 'relevant' with respect to the sequentiality indices of smaller prefixes.

The definition of the set $\mathit{pattern}'_\mathcal{R}(s)$, as it occurs in the Figure 8.1, is loosely based on the *nodes* function as defined by Hanus, Lucas, and Middeldorp [HLM98]. The *nodes* function assigns a definitional tree to a strongly sequential orthogonal constructor TRS.

By the first clause of $\mathit{pattern}'_\mathcal{R}(s)$ and the fact $\mathcal{I}(s)$ is non-empty by Proposition 8.1.10, it is immediate that each term in $\mathit{pattern}_\mathcal{R}$ is a prefix of a rewrite rule in

$$pattern_{\mathcal{R}} = \bigcup_{f \in \mathcal{D}} pattern'_{\mathcal{R}}(f(\bot, \ldots, \bot))$$

$$pattern'_{\mathcal{R}}(s) = \begin{cases} \{v(s)[\bot]_p \mid p \in \mathcal{I}(s)\} & \text{if } s \text{ not a redex in } \mathcal{R} \\ \quad \cup \bigcup_{t \in pattern^*_{\mathcal{R}}(s)} pattern'_{\mathcal{R}}(t) & \\ \emptyset & \text{otherwise} \end{cases}$$

$$pattern^*_{\mathcal{R}}(s) = \{s[f(\bot, \ldots \bot)]_p \preccurlyeq l \mid p \in \mathcal{I}(s) \text{ and } l \to r \in R\}$$

Figure 8.1. The set $pattern_{\mathcal{R}}$

$\mathcal{R}$ with each $\bot$, except for *one*, replaced by a unique fresh variable. By the same facts, and the fact that $\mathcal{I}(s)$ is always finite, it also holds that $pattern_{\mathcal{R}}$ is finite in case $\mathcal{R}$ has a finite number of rewrite rules.

The following holds for $\mathcal{M}$:

Proposition 8.2.2. *The TRS $\mathcal{M}$ is a constructor TRS whose set of defined symbols is a subset of the set of defined symbols of $\mathcal{R}$.*

Proof. Each term in $pattern_{\mathcal{R}}$ is derived from a prefix of a rewrite rule in $\mathcal{R}$, which cannot be equal to $\bot$, by replacing each $\bot$, except for one, by a fresh variable. Hence, we have for each term in $pattern_{\mathcal{R}}$ that each symbol at the root is a defined symbol of $\mathcal{R}$ and that each symbol which does not occur at the root is either a constructor of $\mathcal{R}$, a variable, or $\bot$. Hence, the result is immediate by definition of $\mathcal{M}$ and its dependence on $pattern_{\mathcal{R}}$. □

By the above, we have that the root symbol of the left-hand side of each rewrite rule in $\mathcal{M}$ is a defined symbol of $\mathcal{R}$. Hence, the Böhm-like tree based on $\mathcal{M}$ satisfies the definition of a sufficient part of the produced output, as redexes of $\mathcal{M}$ occur in prefixes consisting solely of constructor symbols.

Example 8.2.3. Assume for the moment that $\mathcal{R}$ consists of the following two rules:

$$f(g(a_1), b_1, x) \to c_1$$
$$f(g(a_2), x, b_2) \to c_2$$

Obviously, $\mathcal{R}$ is an orthogonal constructor TRS. That $\mathcal{R}$ is also strongly sequential is readily proved with the help of Proposition 8.1.10. The following sets of sequentiality indices are relevant for $pattern_{\mathcal{R}}$:

$$\mathcal{I}(f(\bot, \bot, \bot)) = \{1\} \qquad \mathcal{I}(f(g(a_1), \bot, \bot)) = \{2\}$$
$$\mathcal{I}(f(g(\bot), \bot, \bot)) = \{11\} \qquad \mathcal{I}(f(g(a_2), \bot, \bot)) = \{3\}$$

The rewrite rules of $\mathcal{M}$ are now as follows:

$$f(\bot, x, y) \to_\omega \bot \qquad f(g(a_1), \bot, y) \to_\omega \bot$$
$$f(g(\bot), x, y) \to_\omega \bot \qquad f(g(a_2), x, \bot) \to_\omega \bot$$

As can be seen in the above example, there are prefixes of left-hand sides of the rules of $\mathcal{R}$ for which there are no rules in $\mathcal{M}$. This, e.g., holds for $f(\bot, b_1, \bot) \preccurlyeq f(g(a_1), b_1, x)$ and $f(\bot, \bot, b_2) \preccurlyeq f(g(a_2), x, b_2)$. Omitting the rules derived from

these prefixes possibly gives rise to a Böhm-like tree which is larger with respect to the prefix order. Hence, omission is favourable with respect to representing a sufficient part of the produced output.

Of course, we still need to prove that $\mathcal{M}$ is an ω_{m}TRS. To facilitate the proof, we first prove three lemmas. The first concerns the overlap of $\mathcal{R}$ and $\mathcal{M}$. The other two relate sequentiality indices with the definition of $pattern_{\mathcal{R}}$.

Lemma 8.2.4. *The rules of $\mathcal{R}$ and $\mathcal{M}$ do not overlap.*

Proof. By Proposition 8.2.2, overlap can only occur at the root. Thus, suppose the left-hand sides of $l \to r \in R$ and $d \to e \in M$ can be unified. By definition of $pattern'_{\mathcal{R}}$, we have that $d = \upsilon(s)[\bot]_p$ for some $s \in \mathcal{T}er(\Sigma_\bot, V)$ and $p \in \mathcal{I}(s)$. Moreover, by definition of $pattern'_{\mathcal{R}}$ and Proposition 8.1.9, we have for all $d|_q \in V$ that $s|_q = \bot$. Hence, $\sigma(l) = \sigma(d) \not\succcurlyeq s$ and $\sigma(l)|_p = \bot$. As $\bot$ cannot occur in l, there exists a position $p' \leqslant p$ such that $l|_{p'} \in V$. But then, by Proposition 8.1.9, we have $p \notin \mathcal{I}(s)$, a contradiction. Hence, the rules of $\mathcal{R}$ and $\mathcal{M}$ do not overlap. □

Lemma 8.2.5. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$. If $\mathcal{I}(s) \neq \emptyset$ and $s \preccurlyeq t$, then precisely one of the following holds:*

- *there exist $p \in \mathcal{I}(s)$ and substitutions σ such that $t = \sigma(\upsilon(s)[\bot]_p)$, or*
- *$s[root(t|_p)(\bot, \ldots, \bot)]_p \preccurlyeq t$ for all $p \in \mathcal{I}(s)$.*

Proof. Suppose $\mathcal{I}(s) \neq \emptyset$ and $s \preccurlyeq t$. By definition of s and t there are two possibilities: Either there exist $p \in \mathcal{I}(s)$ such that $t|_p = \bot$, or not.

In the first case, it follows immediately that there exists for each $p \in \mathcal{I}(s)$ with $t|_p = \bot$ a substitution σ such that $t = \sigma(\upsilon(s)[\bot]_p)$. In the second case, it is immediate for all $p \in \mathcal{I}(s)$ that $s[root(t|_p)(\bot, \ldots, \bot)]_p \preccurlyeq t$. □

Lemma 8.2.6. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$ such that $\mathcal{I}(s) \neq \emptyset$, $\mathcal{V}ar(s) = \emptyset$, and $s \prec t$. There exist $p \in \mathcal{I}(s)$ with $root(t|_p) \notin V$, if it is assumed that:*

- *$t \preccurlyeq d$ for some $d \in pattern_{\mathcal{R}}$, or*
- *$t \preccurlyeq l$ for some $l \to r$ and t not an $l \to r$-redex.*

Proof. There are two cases to consider: Either there exist $p \in \mathcal{I}(s)$ such that $t|_p = \bot$ or it holds for all $p \in \mathcal{I}(s)$ that $t|_p \neq \bot$. In the first case, the result is immediate. In the second case, suppose for all $p \in \mathcal{I}(s)$ that $root(t|_p) \in V$.

If $t \preccurlyeq d$, then, by definition of $pattern_{\mathcal{R}}$ and Proposition 8.1.9, there exist $s' \in \mathcal{T}er(\Sigma_\bot, V)$, with $\mathcal{V}ar(s') = \emptyset$, and $q \in \mathcal{I}(s')$ such that $d = \upsilon(s')[\bot]_q$. Moreover, since $s \prec t$ and since no variables occur in s, we have $s \prec s'$. Hence, as $p \in \mathcal{I}(s)$ implies $root(t|_p) \in V$ and as $\mathcal{V}ar(s') = \emptyset$, it holds for all $p \in \mathcal{I}(s)$ that $s'|_p = \bot$. By Proposition 8.1.8, $\mathcal{I}(s) \subseteq \mathcal{I}(s')$. Moreover, $\mathcal{I}(s) \supseteq \mathcal{I}(s')$. If not, then there exist $t' \preccurlyeq s \preccurlyeq s'$ and $p' \in \mathcal{I}(t')$ such that $t'|_{p'} = s|_{p'} = \bot$ and $p' \notin \mathcal{I}(s)$, by definition of $pattern_{\mathcal{R}}$. However, this is impossible by Proposition 8.1.8. Hence, $\mathcal{I}(s) = \mathcal{I}(s')$. But then, as $p \in \mathcal{I}(s)$ implies $root(t|_p) \in V$, we have $q \notin \mathcal{I}(s')$, contradiction.

If we have $t \preccurlyeq l$ with t not an $l \to r$-redex, under the above assumption, then it holds for all $p \in \mathcal{I}(s)$ that $l|_p \in V$ and it is immediate by Proposition 8.1.9 that $\mathcal{I}(s)$ is empty, again contraction. Hence, there must exist $p \in \mathcal{I}(s)$ such that $root(t|_p) \notin V$, as required. □

We are now in a position to prove the following:

Lemma 8.2.7. *The TRS $\mathcal{M}$ is an ω_{m}TRS (see Definition 6.4.5).*

Proof. That $\mathcal{M}$ is left-linear is immediate by Proposition 8.1.9 and the definition of υ. We prove in turn each of the five clauses that need to be satisfied by ω_{m}TRSs: Thus, suppose $d \to_{\mathrm{m}} e \in M$ and $l \to r \in R$.

1. By definition of $\mathcal{M}$, it is immediate that e is equal to $\bot$.
2. For each term in $pattern_{\mathcal{R}}$ we have $d = f(s_1, \ldots, s_n)$, with f some defined symbol of $\mathcal{R}$. As $\bot$ is not a defined symbol of $\mathcal{R}$, it does not occur as a right-hand side of a rewrite rule in $\mathcal{M}$. Hence, $\bot$ is a normal form of $\mathcal{M}$.
3. Suppose $s \preccurlyeq d$ or $s \preccurlyeq l$ with s not an $l \to r$-redex. In case $s = \bot$, it is immediate that $s \to_{\mathrm{m}}^{=} \bot$. Thus, suppose $s \neq \bot$. We prove $s \to_{\mathrm{m}} \bot$.
   Obviously, $s \neq \bot$ implies $s = f(s_1, \ldots, s_n)$. Define $t = f(\bot, \ldots, \bot)$. By definition of s and t, it follows from Proposition 8.1.10 that $\mathcal{I}(t) \neq \emptyset$. Hence, as $t \preccurlyeq s$, there are two possibilities by Lemma 8.2.5: Either there exist $p \in \mathcal{I}(t)$ and σ such that $s = \sigma(\upsilon(t)[\bot]_p)$, or $t[root(s|_p)(\bot, \ldots, \bot)]_p \preccurlyeq s$ for all $p \in \mathcal{I}(t)$. In the first case, it is immediate by definition of $\mathcal{M}$ that $s \to_\omega \bot$. In the second case, define $t' = t[root(s|_p)(\bot, \ldots, \bot)]_p$ for some $p \in \mathcal{I}(t)$ such that $root(s|_p) \notin V$, which exists by Lemma 8.2.6. By definition of s and t', it follows from Proposition 8.1.10 that $\mathcal{I}(t') \neq \emptyset$. Hence, we are back in the above case distinction, which is repeated with larger and larger prefixes of s. As s is finite, eventually only the first case of the case distinction applies and we are done.
4. This is a direct consequence of Lemma 8.2.4.
5. This also is a direct consequence of Lemma 8.2.4. □

Assuming the ω_{r}TRS for $\mathcal{R}$ is denoted by $\mathcal{L} = (\Sigma_\bot, L)$, we prove the main theorem of this section:

Theorem 8.2.8. *The TRS defined as $\mathcal{LM} = (\Sigma_\bot, L \cup M)$ is an ωTRS for $\mathcal{R}$.*

Proof. Immediate by Lemma 8.2.7 and Proposition 6.4.7 □

Hence, $\mathcal{M}$ defines a Böhm-like tree for $\mathcal{R}$.

Remark 8.2.9. It is possible to replace the definition of $pattern'_{\mathcal{R}}(s)$ (see Figure 8.1), by the one in Figure 8.2. In the alternative definition, a subset of $\mathcal{I}(s)$ is employed instead of $\mathcal{I}(s)$ itself. By inspection of the proof of Lemma 8.2.7, with $\mathcal{I}(s)$ appropriately replaced by I, it immediately follows that $\mathcal{M}$ is an ω_{m}TRS even in case the alternative definition of $pattern'_{\mathcal{R}}$ is employed.

Different choices for the set I obviously result in different Böhm-like trees. To avoid such choices we employ $\mathcal{I}(s)$ instead of subsets of $\mathcal{I}(s)$.

8.2.2 Tower of Patterns

Assuming that $\mathcal{R} = (\Sigma, R)$ is a strongly sequential orthogonal constructor TRS, that $\mathcal{L} = (\Sigma_\bot, L)$ is the ω_{r}TRS for $\mathcal{R}$, and that $\mathcal{M} = (\Sigma_\bot, M)$ is the ω_{m}TRS defined in the previous section, we next define *towers of patterns*. We employ these towers

$$pattern'_{\mathcal{R}}(s) = \begin{cases} \{v(s)[\bot]_p \mid p \in I\} \\ \quad \cup \bigcup_{t \in pattern^I_{\mathcal{R}}(s)} pattern'_{\mathcal{R}}(t) & \text{if } s \text{ not a redex in } \mathcal{R}, \\ & \text{and } I \subseteq \mathcal{I}(s) \\ \emptyset & \text{otherwise} \end{cases}$$

$$pattern^I_{\mathcal{R}}(s) = \{s[f(\bot, \ldots \bot)]_p \preccurlyeq l \mid p \in I \text{ and } l \to r \in R\}$$

Figure 8.2. Alternative definition of $pattern'_{\mathcal{R}}$

in the next section to prove that strongly sequential orthogonal constructor TRSs are Böhm-like tree sequential.

Given a term s with $s \to^*_\omega \bot$, a tower of patterns for s represents a reduction from s to $\bot$ with respect to $\mathcal{LM} = (\Sigma_\bot, L \cup M)$. The represented reduction is *minimal* in the sense that removing any of the reduction steps no longer yields a reduction from s to $\bot$, even if substitutions are adapted appropriately.

Towers of patterns are defined as follows:

Definition 8.2.10. *Let $s \in \mathcal{T}er(\Sigma_\bot, V)$ such that s reduces to $\bot$ with respect to the ωTRS $\mathcal{LM}$ (see Theorem 8.2.8) and let $LHS = \{d \mid d \to_\omega e \in L \cup M\}$. A* tower of patterns *for s is a non-empty set:*

$$\{(p_i, d_i) \mid 1 \leqslant i \leqslant n\} \subseteq \mathcal{P}os(s) \times LHS\,,$$

such that for all $1 < i \leqslant n$:

- $d_i \in pattern_{\mathcal{R}}$,
- $p_{i-1} = p_i \cdot q_i$ *with* $d_i|_{q_i} = \bot$, *and*
- $(s[\bot]_{p_{i-1}})|p_i = \sigma_i(d_i)$ *with σ_i a substitution,*

and such that $p_n = \epsilon$ and $s|_{p_1} = \sigma(d_1)$ with σ a substitution.

A tower of patterns is said to start *in $\mathcal{R}$ if $d_1 = l$ with $l \to r \in R$ (or, equivalently, $l \to_{\mathrm{r}} \bot \in L$). Moreover, arbitrary towers are denoted by T and T'.*

Remark that $d_i \in pattern_{\mathcal{R}}$ for all $1 < i \leqslant n$ and *not* for all $1 \leqslant i \leqslant n$. Hence, each d_i with $1 < i \leqslant n$ is the left-hand side of a rule in $\mathcal{M}$, while d_1 may either be a left-hand side of a rule in $\mathcal{L}$ or a rule in $\mathcal{M}$, which implies that the notion of a tower of patterns starting in $\mathcal{R}$ is non-void.

Example 8.2.11. Suppose that $\mathcal{R}$ is the strongly sequential orthogonal constructor TRS which only consist of the following rewrite rule:

$$f(c, c) \to c\,.$$

The ω_{m}TRS for $\mathcal{R}$ defined according to Definition 8.2.1 has the following rewrite rules:

$$\begin{array}{ll} f(x, \bot) \to_{\mathrm{m}} \bot & f(c, \bot) \to_{\mathrm{m}} \bot \\ f(\bot, x) \to_{\mathrm{m}} \bot & f(\bot, c) \to_{\mathrm{m}} \bot \end{array}$$

With respect to the above rewrite rules, a number of reductions are possible starting from the term $f(f(c, \bot), f(\bot, c))$. These reductions are depicted in Figure 8.3,

where the label of each edge denotes the position of the redex contracted in the corresponding reduction step.

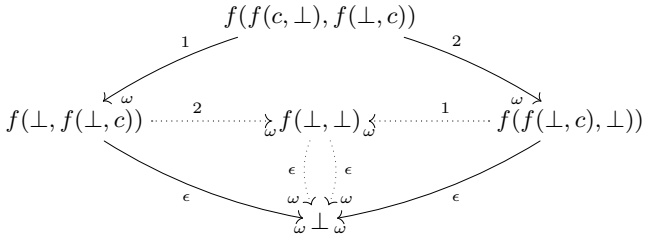

Figure 8.3. Possible reductions of $f(f(c, \bot), f(\bot, c))$

Given the definition of minimality at the beginning of this section, the following are the minimal reductions from Figure 8.3 ending in $\bot$:

$$f(f(c, \bot), f(\bot, c)) \to_\omega f(\bot, f(\bot, c)) \to_\omega \bot$$
$$f(f(c, \bot), f(\bot, c)) \to_\omega f(f(\bot, c), \bot) \to_\omega \bot$$

These two reductions give rise to the following four towers of patterns:

$$\{(1, f(x, \bot)), (\epsilon, f(\bot, x))\} \qquad \{(1, f(c, \bot)), (\epsilon, f(\bot, x))\}$$
$$\{(2, f(\bot, x)), (\epsilon, f(x, \bot))\} \qquad \{(2, f(\bot, c)), (\epsilon, f(x, \bot))\}$$

Hence, each term can have several towers of patterns and minimal reductions ending in $\bot$.

Remark that each term can have both minimal and non-minimal reductions ending in $\bot$. In the case of the reductions depicted in Figure 8.3, any reduction that *does not* proceed along a dotted edge is minimal, while any reduction that *does* proceed along a dotted edge is non-minimal.

With respect to towers of patterns, we obviously have:

Lemma 8.2.12. *Let $s \in \mathcal{T}er(\Sigma_\bot, V)$. The term s reduces to $\bot$ with respect to $\mathcal{LM}$ if and only if there exists a tower of patterns for s.*

Proof. This follows easily by definition of a tower of patterns and the observation that there occurs precisely one $\bot$ in each term of $pattern_\mathcal{R}$. □

The following formalises the intuition behind towers of patterns, where *minimal* is defined as above:

Proposition 8.2.13. *Let $s \in \mathcal{T}er(\Sigma_\bot, V)$ such that s reduces to $\bot$ with respect to $\mathcal{LM}$. If $\{(p_i, d_i) \mid 1 \leqslant i \leqslant n\}$ is a tower of patterns for s, then there exists a non-empty minimal reduction:*

$$s = s_1 \to_{\omega,p_1,d_1} s_2 \to_{\omega,p_2,d_2} \cdots \to_{\omega,p_{n-1},d_{n-1}} s_n \to_{\omega,p_n,d_n} \bot\,,$$

where p_i denotes the position of the contracted redex and where d_i denotes the left-hand side of the employed rewrite rule.

Proof. Given that there occurs precisely one $\bot$ in each term of $pattern_{\mathcal{R}}$, it follows immediately by the definition a tower of patterns that a non-empty reduction exists. That the reduction is minimal follows by the requirement that $p_{i-1} = p_i \cdot q_i$ with $d_i|_{q_i} = \bot$ for all $1 < i \leqslant n$ and the fact that the right-hand side of each rewrite rule in $\mathcal{LM}$ is equal to $\bot$. □

With respect to towers of patterns starting in $\mathcal{R}$, we also have:

Lemma 8.2.14. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$ with $s \preccurlyeq t$. If T is a tower of patterns for s starting in $\mathcal{R}$, then T is a tower of patterns for t starting in $\mathcal{R}$.*

Proof. Suppose that s has a tower of patterns T starting in $\mathcal{R}$. Consider the reduction from Proposition 8.2.13. Since T is a tower of patterns starting in $\mathcal{R}$ we have by the definition of $pattern_{\mathcal{R}}$ that each $\bot$ that occurs in a redex patterns of a contracted redex is created in the reduction. Hence, by left-linearity of $\mathcal{LM}$, the following reduction exists:

$$t = t_1 \to_{\omega,p_1,d_1} t_2 \to_{\omega,p_2,d_2} \cdots \to_{\omega,p_{n-1},d_{n-1}} t_n \to_{\omega,p_n,d_n} \bot\,,$$

which implies that T is a tower of patterns for t. □

Discussion. Towers of patterns merge the notions of a *decomposition* and a *tower of preredexes* as defined by Klop and Middeldorp [KM91, Definition 5.6 and 5.24]. We briefly explain both notions and their connection with towers of patterns. We do not employ the two notions anywhere else in chapter.

In the following we assume that $\mathcal{R} = (\Sigma, R)$ is an orthogonal TRS and that $\mathcal{HL} = (\Sigma_\bot, HL)$ is the Huet-Lévy ωTRS for $\mathcal{R}$. We define decompositions and towers of preredexes:

Definition 8.2.15. *Let*

$$s_1 \to_{\omega,p_1} s_2 \to_{\omega,p_2} \cdots \to_{\omega,p_{n-1}} s_n \to_{\omega,p_n} \bot$$

be a non-empty reduction, where p_i denotes the position of the contracted redex for all $1 \leqslant i \leqslant n$. The set:

$$\{(p_i, s_i|_{p_i}) \mid 1 \leqslant i \leqslant n\}$$

is called a decomposition *of s_1.*

Remark that by the definition of ωTRSs, we have in the previous definition that $p_i \neq p_j$ for all $1 \leqslant i, j \leqslant n$ with $i \neq j$.

Definition 8.2.16. *Let $s \in \mathcal{T}er(\Sigma_\bot, V)$, such that $\omega(s) = \bot$, and let D be a decomposition of s. A non-empty subset D' of D is called a* tower of preredexes *when for all $(p_1, t_1), (p_2, t_2) \in D'$, such that $p_1 \neq p_2$ it holds that:*

- *either $p_1 < p_2$ or $p_2 < p_1$, and*
- *$(q, t) \in D'$, if $(q, t) \in D$ and $p_1 < q < p_2$.*

Changing the definition of the employed pairs slightly, it is readily shown that a tower of patterns merges the definitions a decomposition and a tower of preredexes. Of course, towers of patterns are defined with respect to $\mathcal{LM}$, while decompositions and towers of preredexes are defined with respect to the Huet-Lévy ωTRSs for $\mathcal{R}$.

Assuming that $\mathcal{LM}$ is employed in the above definitions instead of the Huet-Lévy ωTRS, it is immediate by Proposition 8.2.12 that each tower of preredexes is also a decomposition. The same does not hold in case the Huet-Lévy ωTRS is employed. To see this, consider the strongly sequential orthogonal constructor TRS which only has the following rewrite rule:

$$f(c,c) \to c$$

Moreover, consider the following reduction with respect to the Huet-Lévy ωTRS:

$$f(f(c,c),f(c,c)) \to_{\omega,1} f(\bot,f(c,c)) \to_{\omega,2} f(\bot,\bot) \to_{\omega,\epsilon} \bot\,,$$

The decomposition of this reduction is:

$$\{(1,f(c,c)),(2,f(c,c)),(\epsilon,f(\bot,\bot))\}\,.$$

Hence, the towers of preredexes are:

$$\begin{gathered}\{(1,f(c,c))\}\\ \{(2,f(c,c))\}\\ \{(\epsilon,f(\bot,\bot))\}\\ \{(1,f(c,c)),(\epsilon,f(\bot,\bot))\}\\ \{(2,f(c,c)),(\epsilon,f(\bot,\bot))\}\end{gathered}$$

Obviously, none of these towers of preredexes corresponds to a reduction from $f(f(c,c),f(c,c))$ to $\bot$. Hence, none of them defines a decomposition.

8.2.3 Sequentiality

Assuming that $\mathcal{R}$ is a strongly sequential orthogonal constructor TRS and that $\mathcal{M}$ and $\mathcal{LM}$ are, respectively, the ω_{m}TRS and the ωTRS defined in Section 8.2.1, we next show that the Böhm-like tree defined by $\mathcal{LM}$ is Böhm-like tree sequential for $\mathcal{R}$.

In this section, we denote by ω the direct approximant function based on $\mathcal{LM}$. Moreover, given a term s we denote by $\mathcal{T}(s)$ the set containing all towers of patterns for s.

To prove Böhm-like tree sequentiality, we need to consider subterms under reduction. With respect to the subterms we essentially prove that the preservation of towers of patterns starting in $\mathcal{R}$ (see Definition 8.2.10) gives rise to the first clause of the definition of Böhm-like tree sequentiality, while the lack of such towers gives rise to the second clause. To facilitate the proof, we show how towers of patterns are preserved under different circumstances.

The next two lemmas concern towers of patterns starting in $\mathcal{R}$:

Lemma 8.2.17. *Let $s,t \in \mathcal{T}er(\Sigma_\bot,V)$ such that $s \to_{\mathrm{m}} t$, contracting a redex a position p, and such that $s|_p \preccurlyeq \sigma(l)$ and $s|_p$ not an $l \to r$-redex for some $l \to r \in R$ and substitution σ. If for each $s' \in \mathcal{T}er(\Sigma_\bot,V)$ with $s \to^* s'$ there exists a tower of patterns for s' starting in $\mathcal{R}$, then for each $t' \in \mathcal{T}er(\Sigma_\bot,V)$ with $t \to^* t'$ there exists a tower of patterns for t' starting in $\mathcal{R}$.*

Proof. Suppose for each $s \to^* s'$ that there exists a tower of patterns for s' starting in $\mathcal{R}$. Moreover, assume $t \to^* t'$. By definition of ω_{m}TRSs, we have $t = s[\bot]_p \preccurlyeq s$. Hence, by left-linearity of $\mathcal{R}$ there exists a term s' such that $s \to^* s'$ and $t' \preccurlyeq s'$. We prove by contradiction that there is a tower of patterns for t' starting in $\mathcal{R}$.

Suppose there is a tower of patterns T for s' starting in $\mathcal{R}$ which is not a tower of patterns for t' and assume $(p', l') \in T$, $p' \in \mathcal{P}os(s)$, and $l' \to r' \in R$. By Lemma 8.2.14 and as $t = s[\bot]_p$, we have that p' occurs at a position which has as prefix position a descendant q' of p. Since the rules of $\mathcal{R}$ do not overlap and since $s|_p \preccurlyeq \sigma(l)$ and $s|_p$ not an $l \to r$-redex, we have that $s'|_{q'} \preccurlyeq \sigma(l)$ and that there exists a position $q \in \mathcal{P}os(l)$ such that $l|_q \in V$ and $q' < q' \cdot q \leqslant p'$. But then, by Proposition 8.1.9 and the definition of $pattern_{\mathcal{R}}$, it cannot hold that T is tower of patterns for s', contradiction. Hence, each tower of patterns for s' starting in $\mathcal{R}$ is a tower of patterns for t'. □

Lemma 8.2.18. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\bot, V)$ such that $s \neq \bot$, $s \preccurlyeq t$, and $t \to^* t'$. If for each $s' \in \mathcal{T}er(\Sigma_\bot, V)$ with $s \to^* s'$ there exists a tower of patterns for s' starting in $\mathcal{R}$, then there exists a tower of patterns for t' starting in $\mathcal{R}$.*

Proof. Suppose for each $s \to^* s'$ there exists a tower of patterns for s' starting in $\mathcal{R}$. We prove the result for $t \to t'$. The complete result then follows by induction on the number of steps in $t \to^* t'$.

Assuming that the redex contracted in $t \to t'$ occurs at position p, it follows by left-linearity of $\mathcal{R}$ that there are three cases to consider:

– a redex occurs at position p in s,
– the position p does not occur in s, and
– a prefix of a redex occurs at position p in s.

We deal with each of the three cases in turn:

A redex occurs at position p in s. If we contract the redex at position p, then we obtain a term $s' \preccurlyeq t'$. It follows by Lemma 8.2.14 that each tower of patterns for s' starting in $\mathcal{R}$ is also a tower of patterns for t' starting in $\mathcal{R}$, as required.

The position p does not occur in s. Since $s \preccurlyeq t'$, the result follows once more by Lemma 8.2.14.

A prefix of a redex occurs at position p in s. As a prefix of the redex occurs at position p in s, we have $s \to_{\mathrm{m}} s[\bot]_p$ and, obviously, $s[\bot]_p \preccurlyeq t$. The result follows by Lemmas 8.2.17 and 8.2.14. The induction is also furnished by Lemma 8.2.17. □

Supplementing the above two lemmas, the next three concern the preservation of towers of patterns not starting in $\mathcal{R}$:

Lemma 8.2.19. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$ such that $s \neq \bot$, $\omega(s) = \bot$, and $s \to^* t$. If there exist $T \in \mathcal{T}(s)$ not starting in $\mathcal{R}$, then $T \in \mathcal{T}(t)$ and $\omega(t) = \bot$.*

Proof. Suppose there exist $T \in \mathcal{T}(s)$ not starting in $\mathcal{R}$. We prove the result for $s \to t$. The complete result then follows by induction on the number of steps in $s \to^* t$.

Suppose the redex contracted in $s \to t$ occurs at position p. Since T does not start in $\mathcal{R}$, it follows by Proposition 8.2.2 and Lemma 8.2.4 that p cannot be equal

to any $q \cdot q'$ with $(q, d) \in T$, q' in the redex pattern of d. Hence, by left-linearity of $\mathcal{LM}$, we have $T \in \mathcal{T}(t)$. Thus, $\omega(t) = \bot$. □

Lemma 8.2.20. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$ such that $s \neq \bot$, $\omega(s) = \bot$, and $s \to^* t$. If for all $T \in \mathcal{T}(s)$ it holds that T does not start in $\mathcal{R}$, then $\mathcal{T}(s) = \mathcal{T}(t)$.*

Proof. Suppose for all $T \in \mathcal{T}(s)$ that T does not start in $\mathcal{R}$ and assume that the redex contracted in $s \to t$ occurs at position p and employs the rule $l \to r \in R$. We prove the result for $s \to t$. The complete result then follows by induction on the number of steps in $s \to^* t$.

By Lemma 8.2.19, it follows that $\mathcal{T}(s) \subseteq \mathcal{T}(t)$. Hence, suppose that there exist $T \in \mathcal{T}(t)$ such that $T \notin \mathcal{T}(s)$. That is, a tower of patterns is created in the reduction from s to t. As T does not occur in s, there exist $q \cdot q' = p$ with $(q, d) \in T$. Moreover, as towers of patterns are finite sets, there exists a position q which is largest with respect to the prefix order on positions such that $q \cdot q' = p$. With respect to the largest q, we have that $q' \in \mathcal{P}os(d)$, $q' \neq \epsilon$, $d|_{q'} \neq \bot$, and $d|_{q'} \notin V$. Otherwise, either T or a tower of patterns which includes (p, l) occurs in $\mathcal{T}(s)$, which are both impossible by assumption.

Assume that q is the largest position such that $q \cdot q' = p$ and $(q, d) \in T$. Since $d \in \mathit{pattern}_\mathcal{R}$, there exists a term s' such that $d = \upsilon(s')[\bot]_{p'}$ with $p' \in \mathcal{I}(s')$. By definition of $\mathit{pattern}_\mathcal{R}$ and as $d|_{q'} \neq \bot$ and $d|_{q'} \notin V$, there exists a term $t' \preccurlyeq s'[\bot]_{q'} \preccurlyeq d[\bot]_{q'}$ such that $q' \in \mathcal{I}(t')$ and $\upsilon(t')[\bot]_{q'} \in \mathit{pattern}_\mathcal{R}$. Define $d' = \upsilon(t')[\bot]_{q'}$. Moreover, define the following set:

$$T' = \{(p', d) \in T \mid p' < q\} \cup \{(q, d'), (p, l)\}\,.$$

Obviously, we have for some substitutions σ and τ that:

$$s = s[\sigma(l)]_p \to_\omega s[\bot]_p = s[\tau(d')]_q \to_\mathrm{m} s[\bot]_q\,.$$

As $(q, d) \in T$ and as all elements of T that are at prefix positions of q are included in T', we have by left-linearity of $\mathcal{LM}$ that $T' \in \mathcal{T}(s)$. Hence, contradiction, as we assumed that (p, l) cannot occur in any tower of patterns for s. Thus, $\mathcal{T}(s) = \mathcal{T}(t)$. □

Lemma 8.2.21. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$ such that $s \neq \bot$, $\omega(s) = \bot$, and such that for all $T \in \mathcal{T}(s)$ it holds that T does not start in $\mathcal{R}$. If $s \preccurlyeq t$ such that for all $p \in \mathcal{P}os(s)$ with $s|_p = \bot$ it holds that $p \in \mathcal{P}os(t)$, $t|_p \in V \cup \{\bot\}$, and $(q, d) \in T$ with $T \in \mathcal{T}(s)$, $p = q \cdot q'$, and $d|_{q'} = \bot$ implies $t|_p \in V$, then $\omega(t) \neq \bot$.*

Proof. Suppose $s \preccurlyeq t$ such that it satisfies the requirements of the lemma. As is easy to see, t is identical to s with $\bot$ replaced by a variable in a number of cases. By the assumption on t and the fact that T does not start in $\mathcal{R}$ for all $T \in \mathcal{T}(s)$, it follows that none of the towers of patterns of s exists in t.

If $\omega(t) = \bot$, then there exists a tower of patterns T for t by Lemma 8.2.12. However, by the assumptions on t and left-linearity of $\mathcal{LM}$, this implies that T is a tower of pattern for s, contradiction. Hence, $\omega(t) \neq \bot$. □

Denoting by BLT the Böhm-like tree of $\mathcal{R}$ based on $\mathcal{M}$, we next prove the central theorem of this chapter:

Theorem 8.2.22. *The TRS $\mathcal{R}$ is Böhm-like tree sequential with respect to the Böhm-like tree based on $\mathcal{M}$.*

Proof. Let $s \in \mathcal{T}er(\Sigma_\perp, V)$ and $p \in \mathcal{P}os(\mathrm{BLT}(s))$ such that $\mathrm{BLT}(s)|_p = \{\perp\}$. We prove that exactly one of the following two clauses holds, as required by Definition 8.1.1:

1. for all $t \in \mathcal{T}er(\Sigma_\perp, V)$, if $t \not\succcurlyeq s$, then $\mathrm{BLT}(t)|_p = \{\perp\}$, or
2. there exist $q \in \mathcal{P}os(s)$ with $s|_q = \perp$ such that for all terms t, if $t \not\succcurlyeq s$ and $\mathrm{BLT}(t)|_p \neq \{\perp\}$, then $t|_q \neq \perp$.

To see that exactly one of the two clauses holds, remark that there exists a term s' such that $s \to^* s'$, $p \in \mathcal{P}os(s')$, and all subterms at *strict* prefix positions of p root-stable. Hence, by preservation of Böhm-like trees under rewriting, it is enough to consider the reducts of $s'|_p$. With respect to these reducts, which all have $\perp$ as their direct approximant, there are three cases to consider:

1. for each reduct there exists a tower of patterns starting in $\mathcal{R}$,
2. there exists a reduct *without* any tower of patterns starting in $\mathcal{R}$, and
3. the subterm $s'|_p$ is equal to $\perp$.

We deal with each of the three cases in turn:

Case (1). In this case, we prove that the first clause of the definition of Böhm-like tree sequentiality holds. Thus, suppose t is a term such that $t \not\succcurlyeq s$.

By left-linearity of $\mathcal{R}$ there exists a term $t \to^* t'$ such that $t' \not\succcurlyeq s'$. Now consider the subterms $s'|_p$ and $t'|_p$. As there exists for each reduct of $s'|_p$ a tower of patterns starting in $\mathcal{R}$, it follows by Lemma 8.2.18 that there exists for each reduct of $t'|_p$ a tower of patterns starting in $\mathcal{R}$. Hence, each reduct of $t'|_p$ has $\perp$ as its direct approximant. Since we have by monotonicity that $p \in \mathcal{P}os(\mathrm{BLT}(t)) = \mathcal{P}os(\mathrm{BLT}(t'))$, it follows that $\mathrm{BLT}(t)|_p = \mathrm{BLT}(t')|_p = \{\perp\}$, as required by the first clause of the definition of Böhm-like tree sequentiality.

Case (2). In this case, we prove that the second clause of the definition of Böhm-like tree sequentiality holds. Given that $\omega(s'|_p) = \perp$ and that there exists a reduct of $s'|_p$ *without* a tower of patterns starting in $\mathcal{R}$, we can assume without loss of generality that for all $T \in \mathcal{T}(s'|_p)$ it holds that T does not start in $\mathcal{R}$.

Suppose $s' \to^* s''$ and define the following set:

$$P = \{p \cdot q \cdot q' \mid \exists T \in \mathcal{T}(s''|_p) : (q, d) \in T,\ d|_{q'} = \perp,\ \text{and } s''|_{p \cdot q \cdot q'} = \perp\}$$

By Lemma 8.2.20, we have that P is equal for all reducts of s'. Moreover, as $\perp$ does not occur in the rewrite rules of $\mathcal{R}$, there exists a set $P' \subseteq \mathcal{P}os(s)$ such that $P'/(s \to^* s'') = P$ for all $s' \to^* s''$.

Define the term $t \not\succcurlyeq s$ such that for all $p \in \mathcal{P}os(s)$:

$$root(t|_p) = \begin{cases} x \in V & \text{if } p \in P' \\ root(s|_p) & \text{otherwise} \end{cases}$$

Obviously, by left-linearity of $\mathcal{R}$ there exists a term $t \to^* t'$ such that $t' \gtrdot s'$. By Lemma 8.2.21, it now follows that *no* reduct of $t'|_p$ has $\bot$ as its direct approximant. Hence, as we have by monotonicity that $p \in \mathcal{P}os(\mathrm{BLT}(t)) = \mathcal{P}os(\mathrm{BLT}(t'))$, it follows that $\mathrm{BLT}(t)|_p = \mathrm{BLT}(t')|_p \neq \{\bot\}$ and only the second clause of the definition of Böhm-like tree sequentiality can apply, given that one of the clauses applies at all. That the second clause actually holds follows immediately by the fact that P is equal for all $s' \to^* s''$ and Lemma 8.2.19.

Case (3). Identical to the second case, but with $P = \{p\}$, Lemmas 8.2.19 and 8.2.20 replaced by the assumption that $s'|_p = \bot$ and Lemma 8.2.21 replaced by the fact that $\omega(x) = x$ for all $x \in V$. □

Remark 8.2.23. With respect to the proofs presented in this section and the previous section, it is irrelevant whether either the definition of $pattern_{\mathcal{R}}$ from Figure 8.1 or the one from Figure 8.2 is employed. The properties of $pattern_{\mathcal{R}}$ required in the proofs are satisfied by both in both instances.

8.3 Sequentiality of Other Böhm-Like Trees

In this section, we describe two Böhm-like trees which are Böhm-like tree sequential, but which do not represent a sufficient part of the produced output. Throughout this section we assume that $\mathcal{R} = (\Sigma, R)$ is a left-linear TRS.

Trivial Trees. With respect to the trivial trees (see Chapter 5), we have the following:

Theorem 8.3.1. *The TRS $\mathcal{R}$ is Böhm-like tree sequential with respect to trivial trees.*

Proof. Since we have for all $s \in \mathcal{T}er(\Sigma_\bot, V)$ that $\mathrm{BLT}_\mathrm{T}(s) = \{\bot\}$, the first clause of the definition of Böhm-like tree sequentiality always applies. □

Obviously, as each trivial tree is equal to $\{\bot\}$, trivial trees do not represent as sufficient part of the produced output.

Trivial ωTRSs. Suppose $\mathcal{D} = (\Sigma_\bot, D)$, with D defined as:

$$D = \{f(x_1, \ldots, x_n) \to_\omega \bot \mid f(x_1, \ldots x_n) \text{ linear}, f \in \Sigma_n, x_i \in V \text{ for } 1 \leqslant i \leqslant n\}\,.$$

It is easy to show that $\mathcal{D}$ is an ωTRS. We call $\mathcal{D}$ the *trivial ωTRS.*

Given the trivial ωTRS, the Böhm-like tree of a term s is as follows:

$$\mathrm{BLT}(s) = \begin{cases} \{\bot, x\} & \text{if } s \to^* x \text{ with } x \in V \\ \{\bot\} & \text{otherwise} \end{cases}$$

Obviously, these trees do not represent a sufficient part of the produced output. Even though this is the case, Böhm-like trees based on the trivial ωTRS do not always gives rise to Böhm-like tree sequentiality, even if $\mathcal{R}$ is an orthogonal constructor TRS. To see this, consider a Gustave-like orthogonal constructor TRS which has the following three rewrite rules:

$$f(a,b,x,y) \to y$$
$$f(x,a,b,y) \to y$$
$$f(b,x,a,y) \to y$$

The term $f(\bot,\bot,\bot,y)$ is a normal form with respect to the Gustave-like TRS which is not a variable. Hence, we have:

$$\mathrm{BLT}(f(\bot,\bot,\bot,y)) = \{\bot\}\,.$$

Consider the terms $f(a,b,\bot,y)$, $f(\bot,a,b,y)$, and $f(b,\bot,a,y)$. We have:

$$f(a,b,\bot,y) \to y$$
$$f(\bot,a,b,y) \to y$$
$$f(b,\bot,a,y) \to y$$

Moreover, we have the following Böhm-like trees:

$$\mathrm{BLT}(f(a,b,\bot,y)) = \{\bot,y\}$$
$$\mathrm{BLT}(f(\bot,a,b,y)) = \{\bot,y\}$$
$$\mathrm{BLT}(f(b,\bot,a,y)) = \{\bot,y\}$$

Since all three terms have $f(\bot,\bot,\bot,y)$ as prefix and since all three have a Böhm-like tree greater than $\{\bot\}$ with respect to the prefix order, the second clause of the definition of Böhm-like tree sequentiality must apply. However, this is not the case, as the TRS is Gustave-like. Hence, Böhm-like tree sequentiality for $\mathcal{R}$ does not hold with respect to Böhm-like trees based on trivial ωTRSs.

Requiring that $\mathcal{R}$ is confluent and *normal form sequential* changes the above situation:

Theorem 8.3.2. *If $\mathcal{R}$ is confluent and normal form sequential, then $\mathcal{R}$ is Böhm-like tree sequential with respect to Böhm-like trees based on the trivial ωTRS.*

Proof. Suppose $s \in \mathcal{T}er(\Sigma_\bot, V)$ such that $\mathrm{BLT}(s) = \{\bot\}$. There are two possibilities depending on Böhm-like trees of the terms $t \succcurlyeq s$:

For all terms $t \succcurlyeq s$ it holds that $\mathrm{BLT}(t) = \{\bot\}$. In this case it is obvious the first clause of the definition of Böhm-like tree sequentiality applies.

There exist $t \succcurlyeq s$ such that $\mathrm{BLT}(t) = \{\bot, x\}$. In this case, s does not reduce to a normal form in $\mathcal{T}er(\Sigma, V)$. Otherwise, $\mathrm{BLT}(s) = \{\bot, x\}$, by confluence and left-linearity of $\mathcal{R}$ and since the normal form of t is x. Hence, $\mathsf{NF}(s) = \mathsf{F}$.

By the existence of t at least one $\bot$ occurs in s. Hence, as $\mathsf{NF}(t) = \mathsf{T}$, we have by normal form sequentiality of $\mathcal{R}$ and $\mathsf{NF}(s) = \mathsf{F}$ that $\mathcal{I}_{\mathsf{NF}}(s) \neq \emptyset$. But then, it follows for all $t \succcurlyeq s$, that $\mathrm{BLT}(t) = \{\bot, x\}$ implies $t|_p \neq \bot$ for all $p \in \mathcal{I}_{\mathsf{NF}}(s)$. Hence, the second clause of the definition of Böhm-like tree sequentiality holds. □

8.4 Open Problems

With respect to the material presented in the previous sections, a number of interesting open problems exist:

- Is it possible to define a 'sufficient' part of the produced output in a more formal way?
- Besides the Böhm-like tree presented in Section 8.2, can any other Böhm-like tree be defined that is Böhm-like tree sequential for strongly sequential orthogonal constructor TRSs and of which it can be said that sufficient part of the produced output is represented?
- Can the definition of the Böhm-like tree presented in Section 8.2 be extended to all orthogonal (constructor) TRSs in such a way that only the strongly sequential TRSs are Böhm-like tree sequential and such that it can still be said that the Böhm-like trees represent a sufficient part of the produced output?
- Can strong sequentiality, as employed throughout this chapter, be replaced with other forms of sequentiality that give rise to classes of TRSs for which neededness is decidable? In particular, can this be done in the case of shallow sequentiality, as defined by Comon [Com00], and growing sequentiality, as defined by Jacquemard [Jac96].

8.5 Stability

Although Böhm-like tree sequentiality suffices in the first-order case as a definition of non-concurrency, this no longer holds in the higher-order case (see the references in handbook chapter by Ong [Ong95] for relevant literature). In the quest to find a definition of non-concurrency that also suffices in the higher-order case, a number of alternative, but weaker, properties have been formulated. Among these properties is *stability*, as defined by Berry [Ber76].

Given that $\mathcal{R} = (\Sigma, R)$ is a left-linear TRS and that BLT is a *monotone* Böhm-like tree for $\mathcal{R}$, stability is defined as follows:

Definition 8.5.1. *The TRS $\mathcal{R}$ is* stable *with respect to* BLT*, if for every $s \in \mathcal{T}er(\Sigma_\perp, V)$ and $S \in \mathcal{T}er^\infty(\Sigma_\perp, V)$ with $S \preccurlyeq \mathrm{BLT}(s)$ it holds that there exists a unique $t \in \mathcal{T}er(\Sigma_\perp, V)$ which is smallest with respect to the prefix order and which satisfies $t \preccurlyeq s$ and $S \preccurlyeq \mathrm{BLT}(t)$.*

The following theorem relates Böhm-like tree sequentiality and stability, assuming that $\mathcal{R} = (\Sigma, R)$ is a left-linear TRS and that BLT is a monotone Böhm-like tree for $\mathcal{R}$:

Theorem 8.5.2. *If $\mathcal{R}$ is Böhm-like tree sequential with respect to* BLT*, then $\mathcal{R}$ is stable.*

Proof. Suppose $s \in \mathcal{T}er(\Sigma_\perp, V)$ and $S \in \mathcal{T}er^\infty(\Sigma_\perp, V)$. That there exist $t \preccurlyeq s$ such that $S \preccurlyeq \mathrm{BLT}(t)$ is obvious; simply define $t = s$.

Assume there are terms $t_1 \preccurlyeq s$ and $t_2 \preccurlyeq s$ with $t_1 \neq t_2$ such that $S \preccurlyeq \mathrm{BLT}(t_1)$ and $S \preccurlyeq \mathrm{BLT}(t_2)$ and such that *both* are smallest with respect to the prefix order. Since both are smallest, we have for $t_1 \sqcap t_2$ that $S \not\preccurlyeq \mathrm{BLT}(t_1 \sqcap t_2)$. By monotonicity, $\mathrm{BLT}(t_1 \sqcap t_2) \preccurlyeq \mathrm{BLT}(t_1)$ and $\mathrm{BLT}(t_1 \sqcap t_2) \preccurlyeq \mathrm{BLT}(t_2)$. As $S \not\preccurlyeq \mathrm{BLT}(t_1 \sqcap t_2)$, there exist $p \in \mathcal{P}os(\mathrm{BLT}(t_1 \sqcap t_2))$ such that $\mathrm{BLT}(t_1 \sqcap t_2)|_p = \{\perp\}$ and $S|_p \neq \{\perp\}$. Hence,

$\mathrm{BLT}(t_1)|_p \neq \bot$ and $\mathrm{BLT}(t_2)|_p \neq \bot$. But then, by Böhm-like tree sequentiality, there exist $q \in \mathcal{P}os(t_1 \sqcap t_2)$ such that $t_1|_q \neq \bot$, $t_2|_q \neq \bot$, and $(t_1 \sqcap t_2)|_q = \bot$. However, as $t_1 \preccurlyeq s$ and $t_2 \preccurlyeq s$, this implies $t_1|_q = t_2|_q \neq (t_1 \sqcap t_2)|_q$, contradicting the fact that $t_1 \sqcap t_2$ is the greatest lower bound. Hence, there exists a unique $t \in \mathcal{T}er(\Sigma_\bot, V)$ which is smallest with respect to the prefix order and which satisfies $t \preccurlyeq s$ and $S \preccurlyeq \mathrm{BLT}(t)$. □

Obviously, the above proof does not depend on the particular definition of the Böhm-like tree for $\mathcal{R}$. It is sufficient for the Böhm-like tree to be monotone.

9

Higher-Order Rewriting

Curiously enough, the more the obsession with the game grows in the higher dimensions, the less it is actually played, ...

— Douglas Adams
Life, the Universe and Everything (1982)

Böhm-like trees for TRSs, as dealt with in the previous chapters, can be seen as generalisation of the Böhm-like trees of the $\lambda\beta$-calculus. However, to actually call them a generalisation is misleading: The $\lambda\beta$-calculus is a higher-order system, while TRSs are first-order.

To alleviate the above problem, Böhm-like trees for Higher-Order Rewrite Systems (HRSs) are introduced in this chapter. The trees generalise both the Böhm-like trees defined for TRSs and those of $\lambda\beta$-calculus. Besides Böhm-like trees for HRSs, a higher-order analogue of ωTRSs is provided. However, no attempt is made to extend the theory developed in Chapters 6 and 8. This is left as future research.

Most of the theory developed in this chapter straightforwardly extends what was presented in Chapter 5. The extension is straightforward by virtue of the idea underlying direct approximants: the replacement of subterms by some nullary function symbol. We can apply this idea as long as the concept of a subterm exists. Hence, the idea is independent of the considered system being either first-order or higher-order. Of course, the idea does depend of the concept of a term, which is the reason for considering HRSs and not ARSs.

To be able to develop a theory for HRSs which is the analogue of the theory in Chapter 5, we first need to define infinite higher-order terms. Unfortunately, this cannot be done by simply instantiating the approach from Chapter 3: Higher-order terms are typed, while types are completely absent from earlier chapters. For this reason, we only define infinite higher-order terms by means of ideal completion. Ideal completion is a sensible choice, since infinite terms defined in this way are also employed in Chapter 5.

Ideal completion requires the definition of partial higher-order terms and a prefix order on these terms. Defining partial higher-order terms is the only part of the theory developed in this chapter which is not completely straightforward. The most obvious definitions of such terms are problematic with respect to the prefix order: Either no order exists or the order relates terms that should not be related.

Given that the developed theory is the analogue of the theory presented in Chapter 5 and that partial and infinite higher-order terms need to be defined, the current chapter is organised as follows: In Section 9.1, HRSs are introduced. Thereafter, in Sections 9.2 and 9.3, partial higher-order terms and infinite higher-order terms are defined respectively. Sections 9.4, 9.5, and 9.6 follow the organisation

of Chapter 5: In Section 9.4, direct approximant functions and Böhm-like trees are defined. Thereafter, in Section 9.5, monotonicity and continuity are discussed. Finally, in Section 9.6, higher-order ωTRSs are defined.

9.1 Preliminaries

We briefly introduce Higher-Order Rewrite Systems (HRSs). A more complete account of the introduced concepts can be found in Nipkow's paper [Nip91], in the book by Terese [Ter03], and also in the dissertations of Van Oostrom [Oos94] and Van Raamsdonk [Raa96].

To start, assume we have a non-empty set of *base types*. Given these base types, the *(simple) types* are inductively defined as follows:

1. each base type is a (simple) type, and
2. if A and B are (simple) types, then $(A \to B)$ is a (simple) type.

The *type constructor*, denoted $\to$, is assumed to be right associative. In accordance, we leave out parentheses whenever this increases readability. Thus, we write, e.g., $A \to B \to C$ instead of $(A \to (B \to C))$.

Given a type A, the *arity* of A, denoted $ar(A)$, is inductively defined as:

1. if A is a base type, then $ar(A) = 0$,
2. if $A = B \to C$, then $ar(A) = ar(C) + 1$.

We assume for each type A that we have a countably infinite set of *variables* V^A of that type. We write V for $\bigcup\{V^A \mid A \text{ is a type}\}$ and x^A, y^A, z^A, ... for variables of type A. If the type of a variable is clear from the context, or irrelevant, x, y, z, ... are also used.

A *signature* is a set of function symbols, denoted Σ, such that each function symbol has a unique type. We write $f^A, g^A, h^A, \ldots, a^A, b^A, c^A, \ldots$ for function symbols of type A. We also write $f, g, h, \ldots, a, b, c, \ldots$ in case the types of the function symbols are clear from the context or irrelevant. The arity of a function symbol f^A, denoted $ar(f^A)$, is defined as $ar(A)$. A function symbol of arity 0 is called a *nullary* function symbol.

Given a signature Σ and types A and B, *preterms* are inductively defined as:

1. x^A is a preterm, if $x^A \in V^A$,
2. f^A is a preterm, if $f^A \in \Sigma$,
3. $(\lambda x.s)^{A \to B}$ is a preterm, if $x^A \in V^A$ and s^B a preterm, and
4. $(st)^A$ is a preterm, if $s^{B \to A}$ and t^B are preterms.

We write s for a preterm s^A whenever the type of s is clear from the context, or irrelevant. In addition, if $f \in \Sigma$ and $x \in V$ have arity n, then we sometimes write $f(s_1, \ldots, s_n)$ and $x(s_1, \ldots s_n)$ instead of $fs_1 \ldots s_n$ and $xs_1 \ldots s_n$ respectively. Finally, we sometimes use the capitals $F, G, H \ldots, X, Y, Z, \ldots$ to denote free variables, where a variable x is *free* if it does not occur in a subterm of the form $\lambda x.s$.

Remark that preterms are simply typed λ-terms that are defined under the assumption of an additional set of symbols Σ. Following this observation, substitutions, denoted $s[x := t]$, are defined accordingly (see Chapter 4). The notions root symbols, positions, subterms at positions, replacements at positions, and contexts are also defined and denoted accordingly (see Chapters 2 and 3). Of course, types must be taken into account in all definitions.

With respect to preterms, *β-reduction* and *restricted η-expansion* are defined as follows:

$$(\lambda x.s)t \to_\beta s[x := t]$$
$$C[s] \to_{\bar{\eta}} C[\lambda x.sx]$$

where $C[s]$ must satisfy:

1. $s^{A \to B}$ for some types A and B,
2. x is a fresh variable of type A, i.e., it does not occur in $C[s]$,
3. s is not of the form $\lambda y.t$, and
4. $\Box$ does not occur as $(\Box t)$ in $C[\Box]$.

Two observations are important with respect to the above two rules: First, all preterms have a normal form with respect to the rewrite systems consisting of either one or both the above rewrite rules. Second, the set of normal forms with respect to restricted η-expansion is closed under β-reduction. For proofs of both observations see, e.g., the book by Terese [Ter03].

The set of *(higher-order) terms* over the signature Σ, denoted $\mathcal{T}er(\Sigma, V)$, is defined as the set of all preterms in $\beta\bar{\eta}$-normal form. Of course, α-conversion must be taken into account here (see Chapter 3 and Barendregt's book [Bar84]).

Besides the notion of a substitution mentioned above, there is another notion of a substitution which we employ in the definition of HRSs. In this case, a *substitution*, denoted σ, τ, ..., is defined as a map from variables to terms such that the domain is finite and such that the type of each variable and the term assigned to it correspond. If $\sigma = \{x_1 \mapsto t_1, x_2 \mapsto t_2, \ldots, x_n \mapsto t_n\}$, then the *application* of σ to a term s is defined as the $\beta\bar{\eta}$-normal form of

$$(\lambda x_n. \ldots . \lambda x_2.\lambda x_1.s)t_n \ldots t_2 t_1 \,.$$

Observe that a substitution whose domain is a singleton set behaves as expected. That is, it behaves the same as the substitution we defined earlier.

We are now almost in a position to define HRSs, two more concepts are needed. These are as follows:

Definition 9.1.1. *A* pattern *is a term of base type such that each free variable x occurs in a subterm of the form $x(s_1, \ldots, s_n)$ which is of base type and in which each $s_1, \ldots, s_n$ is a $\beta\bar{\eta}$-normal form of a different bound variable. A pattern is called a* rule pattern *if it is of form $f(s_1, \ldots, s_n)$ with $f \in \Sigma$.*

We can now define:

Definition 9.1.2. *A* (higher-order) rewrite rule *is a pair of terms* (l, r) *of the same base type, denoted* $l \to r$*, such that* l *is a rule pattern and such that all free variables that occur in* r *also occur free in* l*.*

A higher-order rewrite system (HRS) *is a pair* $\mathcal{H} = (\Sigma, H)$*, with* Σ *a signature and* H *a set of higher-order rewrite rules.*

We also define:

Definition 9.1.3. *Let* $l \to r$ *be a higher-order rewrite rule. Given a substitution* σ*, the* $\beta\bar{\eta}$*-normal form of* $\sigma(l)$ *is called an* $l \to r$-redex*. If* s *is the* $\beta\bar{\eta}$*-normal form of* $C[\sigma(l)]$ *for some* $l \to r$*-redex and context* $C[\Box]$ *with* $C[\Box]|_p = \Box$*, then an* $l \to r$*-redex, or simply a redex,* occurs *at position* p *and depth* $|p|$ *in* s*. Moreover, if* $q \in \mathcal{P}os(s)$ *then* q *is said to occur in the* redex pattern *of the* $l \to r$ *redex at position* p *in* s*, if* $q \geqslant p$ *and not* $q \geqslant p \cdot p'$ *with* $p' \in \mathcal{P}os(l)$ *such that* $l|_{p'} = x(s_1, \ldots, s_n)$ *with* x *a free variable.*

A pair of terms (s, t)*, denoted* $s \to t$*, defines a* rewrite step*, if* s *and* t *are the* $\beta\bar{\eta}$*-normal forms of respectively* $C[\sigma(l)]$ *and* $C[\sigma(r)]$*, and if* $l \to r$ *is a higher-order rewrite rule. An* $l \to r$*-redex is* contracted *in such a step.*

Root-stable subterms are now defined in the obvious way (see Chapter 2).

Finally, the following definition and example employed later on:

Definition 9.1.4. *A pattern is* linear *if each free variable occurs at most once. A pattern is* fully-extended *if each free variable* x *occurs in a subterm of base type which is of the form* $x(s_1, \ldots, s_n)$ *with* $s_1, \ldots, s_n$ *the* $\beta\bar{\eta}$*-normal forms of all the bound variables.*

An HRS is left-linear*, respectively* fully-extended*, if the left-hand sides of all its rewrite rules are linear, respectively fully-extended.*

Example 9.1.5. Assuming there exists a base type T, it is well-known, see, e.g., the book by Terese [Ter03], that the signature of the λ-calculus can be encoded as the following (higher-order) signature:

$$\{abs^{(T \to T) \to T}, app^{T \to T \to T}\}.$$

Moreover, the β-rule can be encoded as the following (higher-order) rewrite rule:

$$app(abs(\lambda x.F(x)), Y) \to F(Y).$$

9.2 Partial Terms

In this section, we define partial higher-order terms in conjunction with a prefix order on these terms. The definition requires the introduction of a fresh function symbol $\perp$. Since all function symbols in a higher-order signature need to be typed, $\perp$ needs to be typed too.

Although it is very easy to come up with several different typings of $\perp$, typing it in such a way that a prefix order can be defined which satisfies a number of desired properties turns out to be not completely straightforward. Taking into account the definition of the prefix order on partial first-order and partial λ-terms (see

Chapter 3) and also the fact that higher-order terms are typed, there are at least four desired properties:

1. Given that $\bot$ has type A, the partial order must homomorphically extend the requirement that $\bot$ is least among all terms of type A.
2. Terms in which $\bot$ does not occur must be incomparable.
3. Terms of different types must be incomparable.
4. For each type there must be a term that is least among all terms of the type.

The properties help to ensure that the prefix order is a CUSL with respect to each type, which enables us to define for each type a set of infinite terms by means of ideal completion.

To gain more insight in the reason why it is difficult to type $\bot$, we discuss two insufficient typings in Section 9.2.1. The actual typing and prefix order that we employ in the remainder of this chapter are given in Section 9.2.2.

9.2.1 Two Insufficient Typings

We next discuss two typings of $\bot$ that each yield a prefix order which does not satisfy all desired properties mentioned above. As such, the typings can be called insufficient.

First Typing. Possibly the most obvious thing to do, is to assign some arbitrary type to $\bot$. To see that this do not suffice with respect to properties desired of the prefix order, suppose there exist a base type A and two nullary function symbols a_1^A and a_2^A. By the second desirable property, the function symbols must be incomparable. By the fourth property there must exist some term that is a prefix of both function symbols. Since a_1 and a_2 have arity zero, it follows by the first property that only $\bot$ suffices. Hence, by the third property $\bot$ must have type A, not just some arbitrary type.

Suppose there also exist a base type B and two nullary function symbols b_1^B and b_2^B. By repeating the above reasoning, it follows that $\bot$ must have type B. Hence, we can conclude that a single function symbol $\bot$ does not suffice; a number of differently typed functions symbols $\bot$ are needed.

Second Typing. Given the way in which the previous typing fails, we could introduce a function symbol $\bot$ for each type. This has at least four disadvantages, the fourth of which has to do with the prefix order. First, since there are infinitely many types, an infinite number of function symbols will be added to the signature even if the assumed signature is finite. This is not very elegant.

Second, $\bot$ will not only occur as a nullary function symbol, but also as function symbols of arity greater than zero. Assuming we have at our disposal a base type A, we can now specify terms like $\bot(x)$, where $\bot$ is of type $A \to A$ and where $x \in V^A$. Such terms cannot be defined in the first-order case. Hence, to some extent the current typing is incompatible with the first-order approach.

Third, by the assumption that there exists a function symbol $\bot$ for each type we can replace each subterm of a term by $\bot$. However, doing so not always results in a preterm which is also term. That is, the obtained preterm does not

need to be in $\beta\bar{\eta}$-normal form. To see that this is possible, consider the signature $\{c^A, f^{(A\to A)\to A}, g^{A\to A}\}$ and the term $s = f(\lambda x.g(g(c)))$. Replacing the subterm $\lambda x.g(g(c))$ in s by $\bot^{A\to A}$ yields the preterm $f(\bot)$. This preterm is not in $\beta\bar{\eta}$-normal form. The following reduction is possible due to the fact that $\bot$ has type $A \to A$:

$$f(\bot) \to_{\bar{\eta}} f(\lambda x.\bot(x)) .$$

Fourth, given the above signature and terms, it seems obvious that $f(\bot)$ should be a prefix of $s = f(\lambda x.g(g(c)))$, since $f(\bot)$ is obtained by replacing a subterm of s by $\bot$. Whence, as $f(\lambda x.\bot(x))$ is the $\beta\bar{\eta}$-normal form of $f(\bot)$, it also seems obvious that $f(\lambda x.\bot(x))$ should be a prefix of s. However, by the first desirable property for the prefix order, this requires x to be a prefix of either $g(c)$ or c, which violates the second desirable property.

9.2.2 Definition

The problems regarding the typing of $\bot$, as discussed in the previous section, can be avoided by *only* introducing a function symbol $\bot$ for each *base* type. Assuming that Σ is an arbitrary signature and that V is a set of variables, we define:

Definition 9.2.1. *The signature* $\Sigma_\bot$ *is defined as* $\Sigma \cup \{\bot^A \mid A \text{ a base type}\}$. *The set of* partial (higher-order) terms *is defined as the set of higher-order terms over the signature* $\Sigma_\bot$, *i.e.,* $\mathcal{T}er(\Sigma_\bot, V)$.

Given some term in $\mathcal{T}er(\Sigma_\bot, V)$, we always have that some other term in $\mathcal{T}er(\Sigma_\bot, V)$ is obtained whenever some subterm is replaced by $\bot$. The reason is two-fold: First, replacing a subterm by $\bot$ does not introduce a β-redex, since λ-abstractions are not of base type. Second, no $\bar{\eta}$-redex is introduced either, because the $\bar{\eta}$-rule does not apply to terms of base type, like $\bot$.

Given an HRS $\mathcal{H} = (\Sigma, H)$, we can define the HRS $\mathcal{K} = (\Sigma_\bot, H)$. The definition of $\mathcal{K}$ is sound with respect to the rewrite rules of $\mathcal{H}$, as $\Sigma \subseteq \Sigma_\bot$. Moreover, $\mathcal{K}$ has the same termination and confluence properties as $\mathcal{H}$, which follows immediately when we consider each $\bot^A$ to be an appropriately typed variable that is singled out.

Remark 9.2.2. The first two disadvantages of the second typing from Section 9.2.1 do not apply to the above typing: With respect to the first disadvantage, we now have that $\Sigma_\bot$ is infinite only if either Σ or the number of base types is infinite. In any other case $\Sigma_\bot$ is finite.

The second disadvantage does not apply, since each introduced function symbol has arity 0. This implies that $\bot$ cannot occur in a subterm of the form $\bot(x)$.

That the third disadvantage does not apply, is explained below Definition 9.2.1. Moreover, that the problems with respect to the prefix order are also solved, is immediate by Definition 9.2.3 and Lemma 9.2.5 as given below.

Combining the definition of the prefix order on partial first-order and partial λ-terms and taking into account the types, we define the prefix order on partial higher-order preterms as follows:

Definition 9.2.3. *Let Σ be a signature and V a set of variables.*

1. *The* prefix order *on the preterms over $\Sigma_\perp$, denoted $\preccurlyeq$, is the smallest binary relation, modulo α-equivalence, such that:*
   (a) $x^A \preccurlyeq x^A$, if $x \in V^A$,
   (b) $f^A \preccurlyeq f^A$, if $f^A \in \Sigma$,
   (c) $\perp^A \preccurlyeq s^A$, if A is a base type and s^A a preterm,
   (d) $\lambda x.s \preccurlyeq \lambda x.t$, if $x \in V$ and $s \preccurlyeq t$, and
   (e) $s_1 s_2 \preccurlyeq t_1 t_2$, if $s_1 \preccurlyeq t_1$ and $s_2 \preccurlyeq t_2$.
2. *The* strict prefix order *on the preterms over $\Sigma_\perp$, denoted $\prec$, is the smallest binary relation such that for all preterms s and t:*

$$s \prec t \iff (s \preccurlyeq t \text{ and } t \not\preccurlyeq s)\,.$$

If we have $s \preccurlyeq t$, respectively $s \prec t$, then we call s a *prefix* of t, respectively a *strict prefix* of t. Moreover, by $\succcurlyeq$ and $\succ$ we denote respectively the converse of the prefix order and the strict prefix order.

By the explanation just below Definition 9.2.1, it follows immediately that the above definition carries over from preterms to terms without taking into account either β-reduction or restricted η-expansion.

Example 9.2.4. Given the $\lambda\beta$-encoding from Example 9.1.5, we have the following, where s, t_1, and t_2 are arbitrary partial terms:

$$\begin{aligned} \perp &\preccurlyeq abs(\lambda x.s) & app(\perp, t_2) &\preccurlyeq app(t_1, t_2) \\ abs(\lambda x.\perp) &\preccurlyeq abs(\lambda x.s) & app(t_1, \perp) &\preccurlyeq app(t_1, t_2) \\ \perp &\preccurlyeq app(t_1, t_2) & app(\perp, \perp) &\preccurlyeq app(t_1, t_2) \end{aligned}$$

Denoting by $\mathcal{T}er(\Sigma_\perp, V)^A$ all terms of type A in $\mathcal{T}er(\Sigma_\perp, V)$, we have the following lemma with respect to the prefix order and the strict prefix order:

Lemma 9.2.5. *Let A be a type. It holds that the pairs $\mathcal{PO}^A = (\mathcal{T}er(\Sigma_\perp, V)^A, \preccurlyeq)$ and $\mathcal{SPO}^A = (\mathcal{T}er(\Sigma_\perp, V)^A, \prec)$ are respectively a CUSL and a strict partial order.*

Proof. By induction on the structure of preterms, simultaneous for all types, where the least element of type A is:

1. $\perp^A$, if A is a base type, and
2. $\lambda x.l$ with l the least element of type C and $x \in V^B$, if $A = B \to C$.

The proof is completely analogous to the proof of Lemma 3.2.3. □

Like in the first-order case, we have:

Proposition 9.2.6. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$.*

1. *For all $s \preccurlyeq t$ it holds that:*
   – $\mathcal{P}os(s) \subseteq \mathcal{P}os(t)$, and
   – $root(s|_p) = root(t|_p)$, if $p \in \mathcal{P}os(s)$ and $s|_p \neq \perp$.
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 preterms. □

By the above proposition, we also have the following, where the proof is identical to the proof of Proposition 3.2.5:

Proposition 9.2.7. *The strict prefix order on $\mathcal{T}er(\Sigma_\perp, V)$ is well-founded.*

As in the case of partial first-order terms, we can extend the definition of the (strict) prefix order to substitutions by means of a pointwise definition:

Definition 9.2.8. *Let σ and τ be substitutions. Define*

$$\sigma \preccurlyeq \tau \iff (\sigma(x) \preccurlyeq \tau(x) \text{ for all } x \in V)$$

and

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

The prefix order and the strict prefix order on substitutions are respectively a partial order and a strict partial order. This follows immediately by the 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.

We next define root-stable prefixes:

Definition 9.2.9. *Let $\mathcal{H} = (\Sigma, H)$ be an HRS and $s, t \in \mathcal{T}er(\Sigma_\perp, V)$. The term t is a* root-stable prefix *of s, given that $t \preccurlyeq s$ and such that for all $t|_p \neq \perp$ with $p \in \mathcal{P}os(t)$ it holds that $s|_p$ is a root-stable subterm of s.*

We can also have the higher-order analogue of Lemma 5.1.2, which we require in Section 9.6:

Lemma 9.2.10. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$ with t a linear (rule) pattern. If $s \preccurlyeq \tau(t)$ for some substitution τ, then there exist $s' \in \mathcal{T}er(\Sigma_\perp, V)$ and substitutions σ' such that $s = \sigma'(s')$, $s' \preccurlyeq t$, $\sigma' \preccurlyeq \tau$, and s' a linear (rule) pattern.*

Proof. By induction on the number of positions $p \in \mathcal{P}os(s)$ such that $s|_p = \perp$ and $\tau(t)|_p \neq \perp$. Since t is a pattern, this is completely analogous to the proof in first-order case. □

Remark 9.2.11. The previous lemma does not hold in case linear patterns are replaced by linear *terms*. To understand this, suppose $s = f(g(\perp), g(a))$ and $t = F(G)$ and also suppose $\tau(F) = \lambda x.f(x, x)$ and $\tau(G) = g(a)$. Obviously, $\tau(t) = f(g(a), g(a))$ and $s \preccurlyeq \tau(t)$. However, to satisfy the requirements of the lemma, the only choice for s' is $F(G)$. But then, $\sigma'(G)$ must both be equal to $g(\perp)$ and $g(a)$.

9.3 Infinite Terms

For each type, we define the set of infinite terms by means of ideal completion, where we assume Σ to be a signature and V a set of variables:

Definition 9.3.1. *Let A be a type. The set of* infinite (higher-order) terms *of type A, denoted $\mathcal{T}er^\infty(\Sigma_\perp, V)^A$, is defined as:*

$$\mathcal{T}er^\infty(\Sigma_\perp, V)^A = \{I \subseteq \mathcal{T}er(\Sigma_\perp, V)^A \mid I \text{ is an ideal of } \mathcal{PO}^A\}$$

The set of infinite (higher-order) terms*, denoted* $\mathcal{T}er^\infty(\Sigma_\perp, V)$*, is defined as:*

$$\mathcal{T}er^\infty(\Sigma_\perp, V) = \bigcup\{\mathcal{T}er^\infty(\Sigma_\perp, V)^A \mid A \text{ a type}\}\,.$$

Since each $\mathcal{PO}^A$ is a CUSL, we have that $\mathcal{T} = (\mathcal{T}er^\infty(\Sigma_\perp, V)^A, \subseteq)$ is a CPO for each type A, with the least upper bound of a directed set $D \subseteq \mathcal{T}er^\infty(\Sigma_\perp, V)^A$ equal to $\bigcup D$ (see Chapter 2). To achieve notational consistency with (finite) higher-order terms, we write $S \preccurlyeq T$ instead of $S \subseteq T$ and $\bigsqcup D$ instead of $\bigcup D$.

Example 9.3.2. Assuming the encoding of the $\lambda\beta$-calculus from Example 9.1.5, the following are the infinite λ-terms depicted in Figure 3.4 in Section 3.5.1:

$$\downarrow\{abs(\lambda x.abs(\lambda y.abs(\lambda z.\perp))), \ldots\}$$
$$\downarrow\{app(abs(\lambda x.y), app(abs(\lambda x.y), abs(\lambda x.\perp)))\}$$

9.4 Böhm-Like Trees

Having introduced infinite higher-order terms, we next define Böhm-like trees for HRSs. We start with the definition of a direct approximant function, which copies Definition 5.2.13 verbatim:

Definition 9.4.1. *Let* $\mathcal{H} = (\Sigma, H)$ *be a left-linear HRS. A* (higher-order) direct approximant function *for* $\mathcal{H}$ *is a map* $\omega : \mathcal{T}er(\Sigma_\perp, V) \to \mathcal{T}er(\Sigma_\perp, V)$*, such that for all* $s, t, t' \in \mathcal{T}er(\Sigma_\perp, V)$ *and substitutions* σ *it holds that:*

1. $\omega(s) \preccurlyeq s$,
2. *if a redex occurs a position* p *in* s*, then* $\omega(s) \preccurlyeq s[\perp]_p$,
3. *if* $s \to t$*, then* $\omega(s) \preccurlyeq \omega(t)$*, and*
4. *if* $t \,{}^*\!\leftarrow s \to^* t$*, then there exist* s' *such that* $t' \to^* s'$ *and* $\omega(t) \preccurlyeq \omega(s')$.

Note that $s[\perp]_p$ is a well-defined partial term, because each redex and each $\perp$ is of base type.

In the remainder of this section, we assume that $\mathcal{H} = (\Sigma, H)$ is left-linear HRS and that ω is a direct approximant function for $\mathcal{H}$. As in the first-order case, we have the following two lemmas:

Lemma 9.4.2. *If* $s \in \mathcal{T}er(\Sigma_\perp, V)$*, then* $\omega(s)$ *is a root-stable prefix of* s.

Proof. Identical to the proof of Lemma 5.2.2. The proof only depends on the definition of a direct approximant function, which has not been changed. □

Lemma 9.4.3. *Let* $s, t, t' \in \mathcal{T}er(\Sigma_\perp, V)$ *and* $p \in \mathcal{P}os(t) \cap \mathcal{P}os(t')$*. If* $t \,{}^*\!\leftarrow s \to^* t'$ *with* $t|_p$ *and* $t'|_p$ *root-stable and* $root(t|_p) \neq root(t'|_p)$*, then there exist* $q \in \mathcal{P}os(\omega(t))$ *and* $q' \in \mathcal{P}os(\omega(t'))$ *such that* $q, q' \leqslant p$ *and* $\omega(t)|_q = \omega(t')|_{q'} = \perp$.

Proof. Identical to the proof of Lemma 5.2.16. Again, the proof only depends on the definition of a direct approximant function. □

In case $\mathcal{H}$ is confluent, we have that the fourth clause of Definition 9.4.1 follows from the third clause together with confluence:

Proposition 9.4.4. *Let $\mathcal{H}$ be a confluent, left-linear HRS and $s, t, t' \in \mathcal{T}er(\Sigma_\perp, V)$. If $t \,{}^*\!\leftarrow s \to^* t'$, then there exist $s' \in \mathcal{T}er(\Sigma_\perp, V)$ such that $t' \to^* s'$ and $\omega(t) \preccurlyeq \omega(s')$.*

Proof. Identical to the second part of the proof of Proposition 5.2.14. The proof only depends on confluence and the definition of a direct approximant function, which has not been changed. □

We next define auxiliary sets, by copying Definition 5.2.18:

Definition 9.4.5. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. The* auxiliary set *of s (based on ω), denoted $\mathcal{A}(s)$, is defined as:*

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

As in the first-order case and the case of the $\lambda\beta$-calculus, auxiliary sets do not necessarily define infinite terms. The counterexample from the $\lambda\beta$-calculus from Section 4.2 carries over immediately when we employ the encoding of the $\lambda\beta$-calculus from Example 9.1.5. Of course, we do have the following property:

Lemma 9.4.6. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. The set $\mathcal{A}(s)$ is directed.*

Proof. Identical to the proof in Lemma 5.2.19. Again, the proof only depends on the definition of a direct approximant function. □

We next define Böhm-like trees for HRSs:

Definition 9.4.7. *Let $s \in \mathcal{T}er(\Sigma_\perp, V)$. The* Böhm-like tree *of s (based on ω), denoted* $\mathrm{BLT}(s)$*, is defined as:*

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

Analogous to the first-order case, it follows by Lemmas 9.4.2 and 9.4.3 and preservation of root-stability under reduction that a Böhm-like tree only represent root-stable subterms that are shared between different reductions. Moreover, maximal fair reductions are considered by the definition of auxiliary sets.

Again analogous to the first-order case, we have that a Böhm-like tree associates a unique infinite term with a higher-order term. Hence, BLT is a map from $\mathcal{T}er(\Sigma_\perp, V)$ to $\mathcal{T}er^\infty(\Sigma_\perp, V)$.

Böhm-like trees are preserved under rewriting:

Theorem 9.4.8. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$. If $s \to^* t$, then* $\mathrm{BLT}(s) = \mathrm{BLT}(t)$.

Proof. Identical to the proof Theorem 5.2.21. As before, the proof only depends on the definition of a direct approximant function. □

Except for the proofs of Lemmas 9.4.2 and 9.4.3, the proofs given above do not depend on the first and second clause of Definition 9.4.1. They only depend on the other two clauses. Since the other two clauses are independent of any term structure

being present, all proofs except those of Lemmas 9.4.2 and 9.4.3 are easily dealt with in the context of ARSs. This is what is done by Blom [Blo01] and Ariola and Blom [AB02]. Of course, since no terms exist in the context of ARSs, we cannot prove any properties related to root-stability in that context. For this reason, we consider HRSs and not ARSs.

Example 9.4.9. The definitions of trivial trees (Example 5.2.9), normal form trees (Example 5.2.10) and Berarducci-Like trees (Example 5.2.11) carry over immediately to the higher-order setting.

With respect to the $\lambda\beta$-calculus encoding from Example 9.1.5, it is easily verified that the Berarducci-like tree is in fact the Berarducci tree. We have:

$$\begin{aligned}\mathrm{BLT}_{\mathrm{BeL}}(\Delta) &= \downarrow\{\Delta\}\\ \mathrm{BLT}_{\mathrm{BeL}}(\Omega) &= \downarrow\{\bot\}\\ \mathrm{BLT}_{\mathrm{BeL}}(app(\Omega,\Omega)) &= \downarrow\{app(\bot,\bot)\}\end{aligned}$$

where $\Delta = abs(\lambda x.app(x,x))$ and $\Omega = app(\Delta,\Delta)$.

9.5 Monotonicity and Continuity

Given a *fully-extended*, left-linear HRS $\mathcal{H} = (\Sigma, H)$ and a direct approximant function ω for $\mathcal{H}$ that is monotone, we next show that the Böhm-like tree based on ω is monotone and continuous. Fully-extendedness of $\mathcal{H}$ is required in the proof of Lemma 9.5.1. A counterexample in the case fully-extendedness does not hold is provided below. Remember that ω is monotone whenever we have that $s \preccurlyeq t$ implies $\omega(s) \preccurlyeq \omega(t)$.

Lemma 9.5.1. *The Böhm-like tree based on ω is monotone. That is, for all $s, t \in \mathcal{T}er(\Sigma_\bot, V)$, if $s \preccurlyeq t$, then $\mathrm{BLT}(s) \preccurlyeq \mathrm{BLT}(t)$.*

Proof. Analogous to the proof of Lemma 5.3.1. In this case, the term t' exists by left-linearity *and* fully-extendedness. □

Theorem 9.5.2. *The Böhm-like tree based on ω is continuous. That is, if $s \in \mathcal{T}er(\Sigma_\bot, V)$, then $\mathrm{BLT}(s) = \bigsqcup\{\mathrm{BLT}(t) \mid t \preccurlyeq s\}$.*

Proof. Analogous to the proof of Theorem 5.3.2, employing Lemma 9.5.1 instead of Lemma 5.3.1. □

To understand why fully-extendedness is required in the proof of Lemma 9.5.1, consider the HRS which has the following rewrite rule:

$$f(\lambda x.Y) \to c\,.$$

For each term s, define $\omega'(s)$ to be the map that replaces all redexes and all subterms of the form $f(\lambda x.\bot)$ in s by $\bot$. Next, define $\omega(s)$ as the largest prefix of s such that $\omega(s) = \omega'(\omega(s))$. The map ω exists by definition of ω' and Proposition 9.2.7. That ω is monotone follows from the theory developed in the next section.

Consider the terms $f(\lambda x.\bot)$ and $f(\lambda x.x)$. Obviously, $f(\lambda x.\bot) \preccurlyeq f(\lambda x.x)$. However, we also have:

$$\begin{aligned}\mathrm{BLT}(f(\lambda x.\bot)) &= {\downarrow}\{c\}\\ \mathrm{BLT}(f(\lambda x.x)) &= {\downarrow}\{f(\lambda x.x)\}\end{aligned}$$

Whence, we have neither $\mathrm{BLT}(f(\lambda x.\bot)) \preccurlyeq \mathrm{BLT}(f(\lambda x.x))$ nor $\mathrm{BLT}(f(\lambda x.x)) \preccurlyeq \mathrm{BLT}(f(\lambda x.\bot))$, which implies that the Böhm-like tree based on ω is not monotone despite the direct approximant function being monotone.

9.6 Direct Approximant HRSs

We next define a higher-order variant of ωTRSs. Basically copying Definition 5.4.1, the variant is defined as follows:

Definition 9.6.1. *Let $\mathcal{H} = (\Sigma, H)$ be a confluent, left-linear HRS. A direct approximant HRS (ωHRS) for $\mathcal{H}$ is a left-linear HRS $\mathcal{D} = (\Sigma_\bot, D)$, whose rewrite relation, denoted $\to_\omega$, satisfies:*

1. *d is a rule pattern and $e = \bot$ for all $d \to_\omega e \in D$,*
2. *each $\bot$ is a normal form with respect to $\to_\omega$,*
3. *$s \to_\omega^* \bot$ for all $s \preccurlyeq d$ with $d \to_\omega \bot \in D$, and*
4. *$l \to_\omega^* \bot$ for all $l \to r \in R$.*

For each term s in the third clause of the above definition, we have by Lemma 9.2.10 that there must exist a rule pattern t and a substitution τ such that $s = \tau(t)$ and $t \preccurlyeq d$. Hence, it is possible to define for each $s \preccurlyeq d$ a rewrite rule such that $s \to_\omega^{=} \bot$.

Example 9.6.2 (Huet-Lévy ωHRS). The *Huet-Lévy ωHRS* is defined as $\mathcal{HL} = (\Sigma_\bot, HL)$, where $d \to_\omega e \in HL$ if and only if d is a rule pattern, $e = \bot$, and $d \preccurlyeq l$ for some $l \to r \in R$. That the Huet-Lévy ωHRS is actually an ωHRS follows readily from the definition.

With respect to the HRS encoding of the $\lambda\beta$-calculus (see Example 9.1.5), we have that the Huet-Lévy ωHRS consists of the following rewrite rules:

$$\begin{aligned} app(abs(\lambda x.F(x)), Y) &\to_\omega \bot & app(abs(\lambda x.\bot), \bot) &\to_\omega \bot\\ app(abs(\lambda x.F(x)), \bot) &\to_\omega \bot & app(\bot, Y) &\to_\omega \bot\\ app(abs(\lambda x.\bot), Y) &\to_\omega \bot & app(\bot, \bot) &\to_\omega \bot \end{aligned}$$

Employing the transitivity in the third clause of Definition 9.6.1 allows for a slightly more 'economic' ωHRS with the same unique normal forms:

$$\begin{aligned} app(abs(\lambda x.F(x)), Y) &\to_\omega \bot\\ app(\bot, Y) &\to_\omega \bot \end{aligned}$$

Remark that the above two rules form exactly the HRS encoding of the two rules that can be employed to define the Lévy-Longo direct approximant of the $\lambda\beta$-calculus.

Example 9.6.3. Encoding as higher-order rules the rewrite rules that can be employed to define the Böhm direct approximant for the $\lambda\beta$-calculus, we obtain:

$$\begin{aligned} app(abs(\lambda x.F(x)), Y) &\to_\omega \bot \\ abs(\lambda x.\bot) &\to_\omega \bot \\ app(\bot, Y) &\to_\omega \bot \end{aligned}$$

As is easily checked, the rewrite rules form an ωHRS.

In the remainder of this section, we assume that $\mathcal{H} = (\Sigma, H)$ is a confluent, left-linear HRS and that $\mathcal{D} = (\Sigma_\bot, D)$ is an ωHRS for $\mathcal{H}$. Proceeding along the lines of Chapter 5, we next show that each term has a unique normal form with respect to $\mathcal{D}$ and that the map which assigns to each term its unique normal form is a monotone direct approximant function.

To show that each term has a unique normal form, we prove confluence and termination for ωHRSs. To prove confluence, we first consider the case in which the third clause of Definition 9.6.1 can be strengthened to:

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

We call an ωHRS which satisfies the strengthened third clause a *single-step ωHRS.*

Proposition 9.6.4. *If $\mathcal{E} = (\Sigma_\bot, E)$ is a single-step ωHRS, then $\mathcal{E}$ is confluent.*

Proof. Analogous to the proof Proposition 5.4.3. □

Employing single-step ωHRSs, we can now prove:

Lemma 9.6.5. *The ωHRS $\mathcal{D}$ is confluent.*

Proof. Analogous to the proof of Lemma 5.4.4, employing Proposition 9.6.4 instead of Proposition 5.4.3. □

To prove termination, we need the following:

Proposition 9.6.6. *Let $s, t \in \mathcal{T}er(\Sigma_\bot, V)$. If $s \to_\omega t$, then $s \succ t$.*

Proof. Identical to the proof of Proposition 5.4.5. □

We can now prove termination:

Lemma 9.6.7. *The ωHRS $\mathcal{D}$ is terminating.*

Proof. Immediate by Propositions 9.6.6 and 9.2.7. □

By Lemmas 9.6.5 and 9.6.7, we have that each term $s \in \mathcal{T}er(\Sigma_\bot, V)$ has a unique normal form with respect to $\mathcal{D}$. We denote the unique normal form of s by $\omega(s)$.

We next prove that ω defines a monotone direct approximant function. To facilitate the proof, we first prove three lemmas:

Lemma 9.6.8. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\bot, V)$. If $s \preccurlyeq t$ and $t \to_\omega^* t'$, then there exists an $s' \in \mathcal{T}er(\Sigma_\bot, V)$ such that $s' \preccurlyeq t'$ and $s \to_\omega^* s'$.*

Proof. Analogous to the proof of Lemma 5.4.7, employing Lemma 9.2.10 instead of Lemma 5.1.2. □

Lemma 9.6.9. *Let $s, t, t' \in \mathcal{T}er(\Sigma_\perp, V)$. If $s \to^* t$ and $t \to_\omega^* t'$, then there exists an $s' \in \mathcal{T}er(\Sigma_\perp, V)$ such that $s \to_\omega^* s'$ and $s' \preccurlyeq t'$.*

Proof. Analogous to the proof of Lemma 5.4.8, employing Lemma 9.6.8 instead of Lemma 5.4.7. □

Lemma 9.6.10. *Let $s, t \in \mathcal{T}er(\Sigma_\perp, V)$. The following properties hold:*

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

Proof. Analogous to the proof of Lemma 5.4.9, employing Proposition 9.6.6 instead of Proposition 5.4.5 and Lemmas 9.6.8 and 9.6.9 instead of Lemmas 5.4.7 and 5.4.8. □

We can now prove the main theorem of this section:

Theorem 9.6.11. *The map $\omega : \mathcal{T}er(\Sigma_\perp, V) \to \mathcal{T}er(\Sigma_\perp, V)$ which assigns to each term its unique normal form with respect to $\mathcal{D}$ is a monotone direct approximant function.*

Proof. Analogous to the proof of Theorem 5.4.10 employing Lemma 9.6.10 instead of Lemma 5.4.9. □

By the previous theorem, it follows that each ωHRS defines a Böhm-like tree. The tree is monotone and continuous in case the assumed HRS is fully-extended. We have:

Example 9.6.12 (Huet-Lévy Trees). The Huet-Lévy ωHRS of Definition 9.6.2 defines the *Huet-Lévy tree*, denoted $\mathrm{BLT}_{\mathrm{HL}}$.

Huet-Lévy trees are also defined by Blom in his dissertation [Blo01]. However, Blom defines them for Combinatory Reduction Systems instead of HRSs.

The map ω, as defined at the end of the previous section, is easily seen to be definable by the Huet-Lévy ωHRS that consists of the following two rewrite rules:

$$f(\lambda x.Y) \to_\omega \perp$$
$$f(\lambda x.\perp) \to_\omega \perp$$

Hence, the claim that ω defines a monotone direct approximant function is immediate by Theorem 9.6.11.

In case of the $\lambda\beta$-calculus, recall that the Huet-Lévy ωHRS is actually the rewrite system that defines the Lévy-Longo direct approximant. Hence, in the case of the $\lambda\beta$-calculus, Huet-Lévy trees are actually Lévy-Longo trees.

Bibliography

[AB02] Ariola, Z.M., and Blom, S. Skew confluence and the lambda calculus with letrec. *Annals of Pure and Applied Logic 117*, 1–3 (2002), pp. 95–168.

[AC98] Amadio, R.M., and Curien, P.L. *Domains and Lambda-Calculi*, volume 46 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 1998.

[ACG86] Ahuja, S., Carriero, N., and Gelernter, D. Linda and friends. *IEEE Computer 19*, 8 (1986), pp. 26–34.

[AN80] Arnold, A., and Nivat, M. The metric space of infinite trees. Algebraic and topological properties. *Fundamenta Informaticae 3*, 4 (1980), pp. 445–476.

[Ari96] Ariola, Z.M. Relating graph and term rewriting via Böhm models. *Applicable Algebra in Engineering, Communication and Computing 7*, 5 (1996), pp. 401–426.

[Bar84] Barendregt, H.P. *The Lambda Calculus: Its Syntax and Semantics*, revised edition, volume 103 of *Studies in Logic and the Foundations of Mathematics*. Elsevier Science, Amsterdam, 1984.

[Bar93] Barr, M. Terminal coalgebras in well-founded set theory. *Theoretical Computer Science 114*, 2 (1993), pp. 299–315. Corrections and additions can be found in [Bar94]; An updated version can be found on-line [Bar99].

[Bar94] Barr, M. Additions and corrections to "Terminal coalgebras in well-founded set theory". *Theoretical Computer Science 124*, 1 (1994), pp. 189–192.

[Bar99] Barr, M. Terminal coalgebras for endofunctors on sets, 1999. Available from: `ftp://ftp.math.mcgill.ca/pub/barr/pdffiles/trmclg.pdf`.

[Ber76] Berry, G. Bottom-up computations of resursive programs. *Revue Française d'Automatique, Informatique, et Recherche Opérationelle: Informatique Theorique 10*, 3 (1976), pp. 47–82.

[Ber78a] Berry, G. Séquentialité de l'évaluation formelle des λ-expressions. In *Transformations de Programmes: actes du* 3^e *colloque international sur la programmation, Paris, 28-30.03.1978* (1978), Robinet, B., Editor, Dunod, pp. 66–80.

[Ber78b] Berry, G. Stable models of typed λ-calculi. In *Proceedings of the 5th International Colloquium on Automata, Languages and Programming (ICALP'78), Udine, Italy, July 1978* (1978), Ausiello, G., and Böhm, C., Editors, volume 62 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 72–89.

[Ber96] Berarducci, A. Infinite λ-calculus and non-sensible models. In *Logic and Algebra* (1996), Ursini, A., and Aglianò, P., Editors, volume 180 of *Lecture Notes in Pure and Applied Mathematics*, Marcel Dekker, pp. 339–378.

[BL79] Berry, G., and Lévy, J.J. Minimal and optimal computations of recursive programs. *Journal of the ACM 26* (1979), pp. 148–175.

[Blo01] Blom, S.C.C. *Term Graph Rewriting: syntax and semantics*. PhD thesis, Vrije Universiteit, Amsterdam, 2001.

[BN98] Baader, F., and Nipkow, T. *Term Rewriting and All That.* Cambridge University Press, Cambridge, 1998.

[Böh68] Böhm, C. Alcune proprietà delle forme β-η-normali nel λ-K-calcolo. Pubblicazioni dell'Istituto per le Applicazioni del Calcolo N. 696, Roma, 1968.

[Bou85] Boudol, G. Computational semantics of term rewriting systems. In *Algebraic methods in semantics*, Nivat, M., and Reynolds, J.C., Editors. Cambridge University Press, 1985, pp. 169–236.

[Bru72] de Bruijn, N.G. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation with application to the Church-Rosser theorem. *Indagationes Mathematicae 34*, 2 (1972), pp. 381–392.

[BV96] de Bakker, J., and de Vink, E. *Control Flow Semantics.* Foundations of Computing. The MIT Press, Cambridge, Massachusetts, 1996.

[BW99] Barr, M., and Wells, C. *Category Theory for Computing Science*, third edition. Les Publications CRM, Montreal, 1999.

[Chu40] Church, A. A formulation of the simple theory of types. *The Journal of Symbolic Logic 5* (1940), pp. 56–68.

[CN78] Courcelle, B., and Nivat, M. The algebraic semantics of recursive program schemes. In *Proceedings of the 7th Symposium on Mathematical Foundations of Computer Science (MFCS'78), Zakopane, Poland, September 4-8, 1978* (1978), Winkowski, J., Editor, volume 64 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 16–30.

[Com00] Comon, H. Sequentiality, monadic second-order logic and tree automata. *Information and Computation 157*, 1–2 (2000), pp. 25–51.

[Cou79] Courcelle, B. Infinite trees in normal form and recursive equations having a unique solution. *Mathematical Systems Theory 13* (1979), pp. 131–180.

[Cou83] Courcelle, B. Fundamental properties of infinite trees. *Theoretical Computer Science 25*, 2 (1983), pp. 95–169.

[DCSV03] Dezani-Ciancaglini, M., Severi, P., and de Vries, F.J. Infinitary lambda calculus and discrimination of Berarducci trees. *Theoretical Computer Science 298*, 2 (2003), pp. 275–302.

[DKP91] Dershowitz, N., Kaplan, S., and Plaisted, D.A. Rewrite, rewrite, rewrite, rewrite, rewrite, ... *Theoretical Computer Science 83*, 1 (1991), pp. 71–96.

[DM97] Durand, I., and Middeldorp, A. Decidable call by need computations in term rewriting (extended abstract). In *Proceedings of the 14th International Conference on Automated Deduction (CADE-14), Townsville, Australia, 1997* (1997), McCune, W., Editor, volume 1249 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 4–18.

[DP02] Davey, B.A., and Priestley, H.A. *Introduction to Lattices and Order*, second edition. Cambridge University Press, Cambridge, 2002.

[DSW94] Davis, M.D., Sigal, R., and Weyuker, E.J. *Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science*, second edition. Computer Science and Scientific Computing. Academic Press, Boston, 1994.

[FPT99] Fiore, M., Plotkin, G., and Turi, D. Abstract syntax and variable binding (extended abstract). In *Proceedings of the 14th Symposium on Logic in Computer Science (LICS '99), Trento, Italy, July 2-5, 1999* (1999), Computer Society Press of the IEEE, pp. 193–202.

[GLMP03] Ghani, N., Lüth, C., De Marchi, F., and Power, J. Dualizing initial algebras. *Mathematical Structures in Computer Science 13*, 2 (2003), pp. 349–370.

[GP02] Gabbay, M.J., and Pitts, A.M. A new approach to abstract syntax with variable binding. *Formal Aspects of Computing 13* (2002), pp. 341–363.

[GTWW77] Goguen, J.A., Thatcher, J.W., Wagner, E.G., and Wright, J.B. Initial algebra semantics and continuous algebras. *Journal of the ACM 24*, 1 (1977), pp. 68–95.

[Hal60] Halmos, P.R. *Naive Set Theory.* D. Van Nostrand Company, Inc., Princeton, 1960.

[HL91] Huet, G., and Lévy, J.J. Computations in orthogonal rewriting systems, I and II. In *Computational Logic: Essays in honor of Alan Robinson*, Lassez, J.L., and Plotkin, G., Editors. MIT Press, Cambridge, Massachusetts, 1991, pp. 395–414 and 415–443.

[HLM98] Hanus, M., Lucas, S., and Middeldorp, A. Strongly sequential and inductively sequential term rewriting systems. *Information Processing Letters 67*, 1 (1998), pp. 1–8.

[Hug01] Hughes, J. *A Study of Categories of Algebras and Coalgebras.* PhD thesis, Carnegie Mellon University, 2001.

[Hyl75] Hyland, J.M.E. A survey of some useful partial order relations on terms of the lambda calculus. In *Proceedings of the Symposium on λ-calculus and Computer Science Theory, Rome, Italy, March 25-27, 1975* (1975), Böhm, C., Editor, volume 37 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 83–95.

[Hyl76] Hyland, M. A syntactic characterization of the equality in some models for the lambda calculus. *Journal of the London Mathematical Society (2) 12* (1976), pp. 361–370.

[Jac96] Jacquemard, F. Decidable approximations of term rewriting systems. In *Proceedings of the 7th International Conference on Rewriting Techniques and Applications (RTA '96), New Brunswick, USA, 1996* (1996), Ganzinger, H., Editor, volume 1103 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 362–376.

[JP03] Jacobs, B., and Poll, E. Coalgebras and monads in the semantics of Java. *Theoretical Computer Science 291*, 3 (2003), pp. 329–349.

[JR97] Jacobs, B., and Rutten, J. A tutorial on (co)algebras and (co)induction. *Bulletin of EATCS 62* (1997), pp. 222–259.

[Kel75] Kelley, J.L. *General Topology*, volume 27 of *Graduate Texts in Mathematics.* Springer-Verlag, 1975. Reprint of the 1955 edition published by Van Nostrand Company, Inc., Princeton.

[Ket04] Ketema, J. Böhm-like trees for term rewriting systems. In *Proceedings of the 15th International Conference on Rewriting Techniques and Applications (RTA 2004), Aachen, Germany, June 3-5, 2004* (2004), van Oostrom, V., Editor, volume 3091 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 233–248.

[KKSV95] Kennaway, R., Klop, J.W., Sleep, R., and de Vries, F.J. Transfinite reductions in orthogonal term rewriting systems. *Information and Computation 119*, 1 (1995), pp. 18–38.

[KKSV97] Kennaway, J.R., Klop, J.W., Sleep, M., and de Vries, F.J. Infinitary lambda calculus. *Theoretical Computer Science 175*, 1 (1997), pp. 93–125.

[Klo80] Klop, J.W. *Combinatory Reduction Systems.* PhD thesis, Rijksuniversiteit Utrecht, 1980.

[KM91] Klop, J.W., and Middeldorp, A. Sequentiality in orthogonal term rewriting systems. *Journal of Symbolic Computation 12* (1991), pp. 161–195.

[KOV99] Kennaway, R., van Oostrom, V., and de Vries, F.J. Meaningless terms in rewriting. *The Journal of Functional and Logic Programming 1* (1999).

[KP93] Kahn, G., and Plotkin, G. Concrete domains. *Theoretical Computer Science 121*, 1–2 (1993), pp. 187–277.

[Lév75] Lévy, J.J. An algebraic interpretation of the $\lambda\beta$K-calculus and the labelled λ-calculus. In *Proceedings of the Symposium on λ-calculus and Computer Science Theory, Rome, Italy, March 25-27, 1975* (1975), Böhm, C., Editor, volume 37 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 147–165.

[Lév78] Lévy, J.J. *Réductions correctes et optimales dans le lambda-calcul.* PhD thesis, Université de Paris VII, 1978.

[Lin96] Linz, P. *An Introduction to Formal Languages and Automata*, second edition. D. C. Heath and Company, Lexington, Massachusetts, 1996.

[Lon83] Longo, G. Set-theoretical models of λ-calculus: Theories, expansions, isomorphisms. *Annals of Pure and Applied Logic 24* (1983), pp. 153–188.

[Nak75] Nakajima, R. Infinite normal forms for the λ-calculus. In *Proceedings of the Symposium on λ-calculus and Computer Science Theory, Rome, Italy, March 25-27, 1975* (1975), Böhm, C., Editor, volume 37 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 62–82.

[Nip91] Nipkow, T. Higher-order critical pairs. In *Proceedings of the 6th Symposium on Logic in Computer Science (LICS '91), Amsterdam, The Netherlands, July 15-18, 1991* (1991), Computer Society Press of the IEEE, pp. 342–349.

[Ong95] Ong, C.H.L. Correspondence between operational and denotational semantics: The full abstraction problem for PCF. In *Handbook of Logic in Computer Science*, Abramsky, S., Gabbay, D., and Maibaum, T.S.E., Editors, volume 4. Oxford University Press, Oxford, 1995, pp. 269–356.

[Oos94] van Oostrom, V. *Confluence for Abstract and Higher-Order Rewriting.* PhD thesis, Vrije Universiteit, Amsterdam, 1994.

[Oos04] van Oostrom, V. Private communication, 2004.

[Plo77] Plotkin, G.D. LCF considered as a programming language. *Theoretical Computer Science 5* (1977), pp. 223–255.

[Raa96] van Raamsdonk, F. *Confluence and Normalisation for Higher-Order Rewriting.* PhD thesis, Vrije Universiteit, Amsterdam, 1996.

[Rob94] Robinson, E. Variations on algebra: monadicity and generalisations of equational theories. Technical Report 6/94, Sussex Computer Science, 1994.

[Rut98] Rutten, J.J.M.M. Automata and coinduction (an exercise in coalgebra). In *Proceedings of the 9th International Conference on Concurrency Theory (CONCUR'98), Nice, France, September, 1998* (1998), Sangiorgi, D., and de Simone, R., Editors, volume 1466 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 194–217.

[Rut00] Rutten, J.J.M.M. Universal coalgebra: a theory of systems. *Theoretical Computer Science 249*, 1 (2000), pp. 3–80.

[Sco72] Scott, D. Continuous lattices. In *Toposes, Algebraic Geometry and Logic* (1972), Lawvere, F.W., Editor, volume 274 of *Lecture Notes in Mathematics*, Springer-Verlag, pp. 97–136.

[Sco76] Scott, D. Data types as lattices. *SIAM Journal on Computing 5* (1976), pp. 522–587.

[SHLG94] Stoltenberg-Hansen, V., Lindström, I., and Griffor, E.R. *Mathematical Theory of Domains*, volume 22 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 1994.

[Sho67] Shoenfield, J.R. *Mathematical Logic*. Addison-Wesley series in logic. Addison-Wesley, Reading, Massachusetts, 1967.

[SV02a] Severi, P., and de Vries, F.J. An extensional Böhm model. In *Proceedings of the 13th International Conference on Rewriting Techniques and Applications (RTA 2002), Copenhagen, Denmark, July 22-24, 2002* (2002), Tison, S., Editor, volume 2378 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 159–173.

[SV02b] Severi, P., and de Vries, F.J. A lambda calculus for D_∞. Technical Report 2002/29, Department of Mathematics and Computer Sciences, University of Leicester, 2002. Presented at the Workshop on Domain Theory, Copenhagen, Denmark, July 20-21, 2002.

[Tan01] Tanenbaum, A.S. *Modern Operating Systems*, second edition. Prentice Hall, Upper Saddle River, New Jersey, 2001.

[Ter03] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 2003.

[Tur96] Turi, D. *Functorial Operational Semantics and its Denotational Dual*. PhD thesis, Vrije Universiteit, Amsterdam, 1996.

[Vri04] de Vries, F.J. Private communication, 2004.

[Wad76] Wadsworth, C.P. The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus. *SIAM Journal on Computing 5*, 3 (1976), pp. 488–521.

[Wad78] Wadsworth, C.P. Approximate reduction and lambda calculus models. *SIAM Journal on Computing 7*, 3 (1978), pp. 337–356.

[Wel75] Welch, P.H. Continuous semantics and inside-out reductions. In *Proceedings of the Symposium on λ-calculus and Computer Science Theory, Rome, Italy, March 25-27, 1975* (1975), Böhm, C., Editor, volume 37 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 122–146.

[Win93] Winskel, G. *The Formal Semantics of Programming Languages*. Foundations of Computing. The MIT Press, Cambridge, Massachusetts, 1993.

Samenvatting

Je gaat het pas zien als je het doorhebt.

— Johan Cruijff

Böhmbomen voor herschrijfsystemen

In dit proefschrift bestuderen wij denotationele semantiek van *termherschrijfsystemen*. De specifieke semantiek die wij bestuderen is gebaseerd op de syntax van de herschrijfsystemen en is als zodanig afgeleid van de *Böhmboomsemantiek* van de λ-*calculus*. Grofweg betekent dit, dat de denotatie van een term de limiet is van alle termen die voorkomen in alle reducties die beginnen in de term.

Vanuit de literatuur zijn twee methoden bekend om Böhmbomen te construeren: Ten eerste kunnen de bomen geconstrueerd worden door uit iedere term een benadering van zijn Böhmboom te extraheren en door vervolgens de benaderingen van alle reducten van een term te verzamelen. Ten tweede is het mogelijk termherschrijfsystemen uit te breiden met oneindige termen en oneindige reducties en Böhmbomen te definiëren als oneindige normaalvormen.

Het zwaartepunt van dit proefschrift ligt bij de eerste van de twee bovengenoemde methoden. Wij definiëren een aantal beperkingen op de benaderingen die wij toestaan, zodat de Böhmbomen die op de benaderingen gebaseerd zijn een aantal specifieke eigenschappen hebben. De eigenschappen zijn als volgt:

– *Modeleigenschap*: Deze eigenschap houdt ten eerste in dat de Böhmboom van een term identiek is aan de Böhmboom van ieder reduct van deze term. Ten tweede houdt deze eigenschap in, dat als de Böhmbomen van twee termen identiek zijn, deze identiek blijven als de termen ieder in dezelfde context geplaatst worden.
– *Continuïteitseigenschappen*: Deze eigenschappen houden in dat het nemen van een Böhmboom van een term en het plaatsen van een Böhmboom in een context continue operaties zijn in ordetheoretische zin, gegeven een geschikte ordening op termen en Böhmbomen.
– *Sequentialiteitseigenschap*: Deze eigenschap geeft informatie over de sequentialiteit van het beschouwde termherschrijfsysteem.

Behalve het formuleren van een aantal beperkingen zodat aan bovenstaande eigenschappen wordt voldaan, vergelijken wij de benaderingsmethode ook met de methode die Böhmbomen definieert op basis van oneindige termen en oneindige reducties. Hiernaast snijden wij nog een drietal kleinere onderwerpen aan:

– Wij vergelijken drie bekende manieren om oneindige termen te definiëren met behulp van coalgebraïsche technieken.

– Wij geven een overzicht van de vanuit de literatuur bekende Böhmbomen die gedefinieerd zijn voor de λ-calculus en wij catalogiseren enige eigenschappen van deze bomen.
– Wij breiden de basisdefinities van de benaderingsmethode voor Böhmbomen uit naar hogere-orde herschrijfsystemen.

Titles in the IPA Dissertation Series

J.O. Blanco. *The State Operator in Process Algebra*. Faculty of Mathematics and Computing Science, TUE. 1996-01

A.M. Geerling. *Transformational Development of Data-Parallel Algorithms*. Faculty of Mathematics and Computer Science, KUN. 1996-02

P.M. Achten. *Interactive Functional Programs: Models, Methods, and Implementation*. Faculty of Mathematics and Computer Science, KUN. 1996-03

M.G.A. Verhoeven. *Parallel Local Search*. Faculty of Mathematics and Computing Science, TUE. 1996-04

M.H.G.K. Kesseler. *The Implementation of Functional Languages on Parallel Machines with Distrib. Memory*. Faculty of Mathematics and Computer Science, KUN. 1996-05

D. Alstein. *Distributed Algorithms for Hard Real-Time Systems*. Faculty of Mathematics and Computing Science, TUE. 1996-06

J.H. Hoepman. *Communication, Synchronization, and Fault-Tolerance*. Faculty of Mathematics and Computer Science, UvA. 1996-07

H. Doornbos. *Reductivity Arguments and Program Construction*. Faculty of Mathematics and Computing Science, TUE. 1996-08

D. Turi. *Functorial Operational Semantics and its Denotational Dual*. Faculty of Mathematics and Computer Science, VUA. 1996-09

A.M.G. Peeters. *Single-Rail Handshake Circuits*. Faculty of Mathematics and Computing Science, TUE. 1996-10

N.W.A. Arends. *A Systems Engineering Specification Formalism*. Faculty of Mechanical Engineering, TUE. 1996-11

P. Severi de Santiago. *Normalisation in Lambda Calculus and its Relation to Type Inference*. Faculty of Mathematics and Computing Science, TUE. 1996-12

D.R. Dams. *Abstract Interpretation and Partition Refinement for Model Checking*. Faculty of Mathematics and Computing Science, TUE. 1996-13

M.M. Bonsangue. *Topological Dualities in Semantics*. Faculty of Mathematics and Computer Science, VUA. 1996-14

B.L.E. de Fluiter. *Algorithms for Graphs of Small Treewidth*. Faculty of Mathematics and Computer Science, UU. 1997-01

W.T.M. Kars. *Process-algebraic Transformations in Context*. Faculty of Computer Science, UT. 1997-02

P.F. Hoogendijk. *A Generic Theory of Data Types*. Faculty of Mathematics and Computing Science, TUE. 1997-03

T.D.L. Laan. *The Evolution of Type Theory in Logic and Mathematics*. Faculty of Mathematics and Computing Science, TUE. 1997-04

C.J. Bloo. *Preservation of Termination for Explicit Substitution*. Faculty of Mathematics and Computing Science, TUE. 1997-05

J.J. Vereijken. *Discrete-Time Process Algebra*. Faculty of Mathematics and Computing Science, TUE. 1997-06

F.A.M. van den Beuken. *A Functional Approach to Syntax and Typing*. Faculty of Mathematics and Informatics, KUN. 1997-07

A.W. Heerink. *Ins and Outs in Refusal Testing*. Faculty of Computer Science, UT. 1998-01

G. Naumoski and W. Alberts. *A Discrete-Event Simulator for Systems Engineering*. Faculty of Mechanical Engineering, TUE. 1998-02

J. Verriet. *Scheduling with Communication for Multiprocessor Computation*. Faculty of Mathematics and Computer Science, UU. 1998-03

J.S.H. van Gageldonk. *An Asynchronous Low-Power 80C51 Microcontroller*. Faculty of Mathematics and Computing Science, TUE. 1998-04

A.A. Basten. *In Terms of Nets: System Design with Petri Nets and Process Algebra*. Faculty of Mathematics and Computing Science, TUE. 1998-05

E. Voermans. *Inductive Datatypes with Laws and Subtyping – A Relational Model*. Faculty of Mathematics and Computing Science, TUE. 1999-01

H. ter Doest. *Towards Probabilistic Unification-based Parsing*. Faculty of Computer Science, UT. 1999-02

J.P.L. Segers. *Algorithms for the Simulation of Surface Processes*. Faculty of Mathematics and Computing Science, TUE. 1999-03

C.H.M. van Kemenade. *Recombinative Evolutionary Search*. Faculty of Mathematics and Natural Sciences, UL. 1999-04

E.I. Barakova. *Learning Reliability: a Study on Indecisiveness in Sample Selection*. Faculty of Mathematics and Natural Sciences, RUG. 1999-05

M.P. Bodlaender. *Scheduler Optimization in Real-Time Distributed Databases*. Faculty of Mathematics and Computing Science, TUE. 1999-06

M.A. Reniers. *Message Sequence Chart: Syntax and Semantics*. Faculty of Mathematics and Computing Science, TUE. 1999-07

J.P. Warners. *Nonlinear approaches to satisfiability problems*. Faculty of Mathematics and Computing Science, TUE. 1999-08

J.M.T. Romijn. *Analysing Industrial Protocols with Formal Methods*. Faculty of Computer Science, UT. 1999-09

P.R. D'Argenio. *Algebras and Automata for Timed and Stochastic Systems*. Faculty of Computer Science, UT. 1999-10

G. Fábián. *A Language and Simulator for Hybrid Systems*. Faculty of Mechanical Engineering, TUE. 1999-11

J. Zwanenburg. *Object-Oriented Concepts and Proof Rules*. Faculty of Mathematics and Computing Science, TUE. 1999-12

R.S. Venema. *Aspects of an Integrated Neural Prediction System*. Faculty of Mathematics and Natural Sciences, RUG. 1999-13

J. Saraiva. *A Purely Functional Implementation of Attribute Grammars*. Faculty of Mathematics and Computer Science, UU. 1999-14

R. Schiefer. *Viper, A Visualisation Tool for Parallel Program Construction*. Faculty of Mathematics and Computing Science, TUE. 1999-15

K.M.M. de Leeuw. *Cryptology and Statecraft in the Dutch Republic*. Faculty of Mathematics and Computer Science, UvA. 2000-01

T.E.J. Vos. *UNITY in Diversity. A stratified approach to the verification of distributed algorithms*. Faculty of Mathematics and Computer Science, UU. 2000-02

W. Mallon. *Theories and Tools for the Design of Delay-Insensitive Communicating Processes*. Faculty of Mathematics and Natural Sciences, RUG. 2000-03

W.O.D. Griffioen. *Studies in Computer Aided Verification of Protocols*. Faculty of Science, KUN. 2000-04

P.H.F.M. Verhoeven. *The Design of the MathSpad Editor*. Faculty of Mathematics and Computing Science, TUE. 2000-05

J. Fey. *Design of a Fruit Juice Blending and Packaging Plant*. Faculty of Mechanical Engineering, TUE. 2000-06

M. Franssen. *Cocktail: A Tool for Deriving Correct Programs*. Faculty of Mathematics and Computing Science, TUE. 2000-07

P.A. Olivier. *A Framework for Debugging Heterogeneous Applications*. Faculty of Natural Sciences, Mathematics and Computer Science, UvA. 2000-08

E. Saaman. *Another Formal Specification Language*. Faculty of Mathematics and Natural Sciences, RUG. 2000-10

M. Jelasity. *The Shape of Evolutionary Search Discovering and Representing Search Space Structure*. Faculty of Mathematics and Natural Sciences, UL. 2001-01

R. Ahn. *Agents, Objects and Events a computational approach to knowledge, observation and communication*. Faculty of Mathematics and Computing Science, TU/e. 2001-02

M. Huisman. *Reasoning about Java programs in higher order logic using PVS and Isabelle*. Faculty of Science, KUN. 2001-03

I.M.M.J. Reymen. *Improving Design Processes through Structured Reflection*. Faculty of Mathematics and Computing Science, TU/e. 2001-04

S.C.C. Blom. *Term Graph Rewriting: syntax and semantics*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2001-05

R. van Liere. *Studies in Interactive Visualization*. Faculty of Natural Sciences, Mathematics and Computer Science, UvA. 2001-06

A.G. Engels. *Languages for Analysis and Testing of Event Sequences*. Faculty of Mathematics and Computing Science, TU/e. 2001-07

J. Hage. *Structural Aspects of Switching Classes*. Faculty of Mathematics and Natural Sciences, UL. 2001-08

M.H. Lamers. *Neural Networks for Analysis of Data in Environmental Epidemiology: A Case-study into Acute Effects of Air Pollution Episodes*. Faculty of Mathematics and Natural Sciences, UL. 2001-09

T.C. Ruys. *Towards Effective Model Checking*. Faculty of Computer Science, UT. 2001-10

D. Chkliaev. *Mechanical verification of concurrency control and recovery protocols*. Faculty of Mathematics and Computing Science, TU/e. 2001-11

M.D. Oostdijk. *Generation and presentation of formal mathematical documents*. Faculty of Mathematics and Computing Science, TU/e. 2001-12

A.T. Hofkamp. *Reactive machine control: A simulation approach using χ*. Faculty of Mechanical Engineering, TU/e. 2001-13

D. Bošnački. *Enhancing state space reduction techniques for model checking*. Faculty of Mathematics and Computing Science, TU/e. 2001-14

M.C. van Wezel. *Neural Networks for Intelligent Data Analysis: theoretical and experimental aspects*. Faculty of Mathematics and Natural Sciences, UL. 2002-01

V. Bos and J.J.T. Kleijn. *Formal Specification and Analysis of Industrial Systems*. Faculty of Mathematics and Computer Science and Faculty of Mechanical Engineering, TU/e. 2002-02

T. Kuipers. *Techniques for Understanding Legacy Software Systems*. Faculty of Natural Sciences, Mathematics and Computer Science, UvA. 2002-03

S.P. Luttik. *Choice Quantification in Process Algebra*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2002-04

R.J. Willemen. *School Timetable Construction: Algorithms and Complexity*. Faculty of Mathematics and Computer Science, TU/e. 2002-05

M.I.A. Stoelinga. *Alea Jacta Est: Verification of Probabilistic, Real-time and Parametric Systems*. Faculty of Science, Mathematics and Computer Science, KUN. 2002-06

N. van Vugt. *Models of Molecular Computing*. Faculty of Mathematics and Natural Sciences, UL. 2002-07

A. Fehnker. *Citius, Vilius, Melius: Guiding and Cost-Optimality in Model Checking of Timed and Hybrid Systems*. Faculty of Science, Mathematics and Computer Science, KUN. 2002-08

R. van Stee. *On-line Scheduling and Bin Packing*. Faculty of Mathematics and Natural Sciences, UL. 2002-09

D. Tauritz. *Adaptive Information Filtering: Concepts and Algorithms*. Faculty of Mathematics and Natural Sciences, UL. 2002-10

M.B. van der Zwaag. *Models and Logics for Process Algebra*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2002-11

J.I. den Hartog. *Probabilistic Extensions of Semantical Models*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2002-12

L. Moonen. *Exploring Software Systems*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2002-13

J.I. van Hemert. *Applying Evolutionary Computation to Constraint Satisfaction and Data Mining*. Faculty of Mathematics and Natural Sciences, UL. 2002-14

S. Andova. *Probabilistic Process Algebra*. Faculty of Mathematics and Computer Science, TU/e. 2002-15

Y.S. Usenko. *Linearization in μCRL*. Faculty of Mathematics and Computer Science, TU/e. 2002-16

J.J.D. Aerts. *Random Redundant Storage for Video on Demand*. Faculty of Mathematics and Computer Science, TU/e. 2003-01

M. de Jonge. *To Reuse or To Be Reused: Techniques for component composition and construction*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2003-02

J.M.W. Visser. *Generic Traversal over Typed Source Code Representations*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2003-03

S.M. Bohte. *Spiking Neural Networks*. Faculty of Mathematics and Natural Sciences, UL. 2003-04

T.A.C. Willemse. *Semantics and Verification in Process Algebras with Data and Timing*. Faculty of Mathematics and Computer Science, TU/e. 2003-05

S.V. Nedea. *Analysis and Simulations of Catalytic Reactions*. Faculty of Mathematics and Computer Science, TU/e. 2003-06

M.E.M. Lijding. *Real-time Scheduling of Tertiary Storage*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2003-07

H.P. Benz. *Casual Multimedia Process Annotation – CoMPAs*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2003-08

D. Distefano. *On Modelchecking the Dynamics of Object-based Software: a Foundational Approach*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2003-09

M.H. ter Beek. *Team Automata – A Formal Approach to the Modeling of Collaboration Between System Components*. Faculty of Mathematics and Natural Sciences, UL. 2003-10

D.J.P. Leijen. *The λ Abroad – A Functional Approach to Software Components*. Faculty of Mathematics and Computer Science, UU. 2003-11

W.P.A.J. Michiels. *Performance Ratios for the Differencing Method*. Faculty of Mathematics and Computer Science, TU/e. 2004-01

G.I. Jojgov. *Incomplete Proofs and Terms and Their Use in Interactive Theorem Proving*. Faculty of Mathematics and Computer Science, TU/e. 2004-02

P. Frisco. *Theory of Molecular Computing – Splicing and Membrane systems*. Faculty of Mathematics and Natural Sciences, UL. 2004-03

S. Maneth. *Models of Tree Translation*. Faculty of Mathematics and Natural Sciences, UL. 2004-04

Y. Qian. *Data Synchronization and Browsing for Home Environments*. Faculty of Mathematics and Computer Science and Faculty of Industrial Design, TU/e. 2004-05

F. Bartels. *On Generalised Coinduction and Probabilistic Specification Formats*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2004-06

L. Cruz-Filipe. *Constructive Real Analysis: a Type-Theoretical Formalization and Applications*. Faculty of Science, Mathematics and Computer Science, KUN. 2004-07

E.H. Gerding. *Autonomous Agents in Bargaining Games: An Evolutionary Investigation of Fundamentals, Strategies, and Business Applications*. Faculty of Technology Management, TU/e. 2004-08

N. Goga. *Control and Selection Techniques for the Automated Testing of Reactive Systems*.

Faculty of Mathematics and Computer Science, TU/e. 2004-09

M. Niqui. *Formalising Exact Arithmetic: Representations, Algorithms and Proofs*. Faculty of Science, Mathematics and Computer Science, RU. 2004-10

A. Löh. *Exploring Generic Haskell*. Faculty of Mathematics and Computer Science, UU. 2004-11

I.C.M. Flinsenberg. *Route Planning Algorithms for Car Navigation*. Faculty of Mathematics and Computer Science, TU/e. 2004-12

R.J. Bril. *Real-time Scheduling for Media Processing Using Conditionally Guaranteed Budgets*. Faculty of Mathematics and Computer Science, TU/e. 2004-13

J. Pang. *Formal Verification of Distributed Systems*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2004-14

F. Alkemade. *Evolutionary Agent-Based Economics*. Faculty of Technology Management, TU/e. 2004-15

E.O. Dijk. *Indoor Ultrasonic Position Estimation Using a Single Base Station*. Faculty of Mathematics and Computer Science, TU/e. 2004-16

S.M. Orzan. *On Distributed Verification and Verified Distribution*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2004-17

M.M. Schrage. *Proxima - A Presentation-oriented Editor for Structured Documents*. Faculty of Mathematics and Computer Science, UU. 2004-18

E. Eskenazi and A. Fyukov. *Quantitative Prediction of Quality Attributes for Component-Based Software Architectures*. Faculty of Mathematics and Computer Science, TU/e. 2004-19

P.J.L. Cuijpers. *Hybrid Process Algebra*. Faculty of Mathematics and Computer Science, TU/e. 2004-20

N.J.M. van den Nieuwelaar. *Supervisory Machine Control by Predictive-Reactive Scheduling*. Faculty of Mechanical Engineering, TU/e. 2004-21

E. Ábrahám. *An Assertional Proof System for Multithreaded Java -Theory and Tool Support-*. Faculty of Mathematics and Natural Sciences, UL. 2005-01

R. Ruimerman. *Modeling and Remodeling in Bone Tissue*. Faculty of Biomedical Engineering, TU/e. 2005-02

C.N. Chong. *Experiments in Rights Control - Expression and Enforcement*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2005-03

H. Gao. *Design and Verification of Lock-free Parallel Algorithms*. Faculty of Mathematics and Computing Sciences, RUG. 2005-04

H.M.A. van Beek. *Specification and Analysis of Internet Applications*. Faculty of Mathematics and Computer Science, TU/e. 2005-05

M.T. Ionita. *Scenario-Based System Architecting - A Systematic Approach to Developing Future-Proof System Architectures*. Faculty of Mathematics and Computing Sciences, TU/e. 2005-06

G. Lenzini. *Integration of Analysis Techniques in Security and Fault-Tolerance*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2005-07

I. Kurtev. *Adaptability of Model Transformations*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2005-08

T. Wolle. *Computational Aspects of Treewidth - Lower Bounds and Network Reliability*. Faculty of Science, UU. 2005-09

O. Tveretina. *Decision Procedures for Equality Logic with Uninterpreted Functions*. Faculty of Mathematics and Computer Science, TU/e. 2005-10

A.M.L. Liekens. *Evolution of Finite Populations in Dynamic Environments*. Faculty of Biomedical Engineering, TU/e. 2005-11

J. Eggermont. *Data Mining using Genetic Programming: Classification and Symbolic Regression*. Faculty of Mathematics and Natural Sciences, UL. 2005-12

B.J. Heeren. *Top Quality Type Error Messages*. Faculty of Science, UU. 2005-13

G.F. Frehse. *Compositional Verification of Hybrid Systems using Simulation Relations*. Faculty of Science, Mathematics and Computer Science, RU. 2005-14

M.R. Mousavi. *Structuring Structural Operational Semantics*. Faculty of Mathematics and Computer Science, TU/e. 2005-15

A. Sokolova. *Coalgebraic Analysis of Probabilistic Systems*. Faculty of Mathematics and Computer Science, TU/e. 2005-16

T. Gelsema. *Effective Models for the Structure of pi-Calculus Processes with Replication*. Faculty of Mathematics and Natural Sciences, UL. 2005-17

P. Zoeteweij. *Composing Constraint Solvers*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2005-18

J.J. Vinju. *Analysis and Transformation of Source Code by Parsing and Rewriting*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2005-19

M. Valero Espada. *Modal Abstraction and Replication of Processes with Data*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2005-20

A. Dijkstra. *Stepping through Haskell*. Faculty of Science, UU. 2005-21

Y.W. Law. *Key management and link-layer security of wireless sensor networks: energy-efficient attack and defense*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2005-22

E. Dolstra. *The Purely Functional Software Deployment Model*. Faculty of Science, UU. 2006-01

R.J. Corin. *Analysis Models for Security Protocols*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2006-02

P.R.A. Verbaan. *The Computational Complexity of Evolving Systems*. Faculty of Science, UU. 2006-03

K.L. Man and R.R.H. Schiffelers. *Formal Specification and Analysis of Hybrid Systems*. Faculty of Mathematics and Computer Science and Faculty of Mechanical Engineering, TU/e. 2006-04

M. Kyas. *Verifying OCL Specifications of UML Models: Tool Support and Compositionality*. Faculty of Mathematics and Natural Sciences, UL. 2006-05

M. Hendriks. *Model Checking Timed Automata - Techniques and Applications*. Faculty of Science, Mathematics and Computer Science, RU. 2006-06

J. Ketema. *Böhm-Like Trees for Rewriting*. Faculty of Sciences, VUA. 2006-07