A Fully Abstract Denotational Model for Observational Precongruence

Anna Ingólfsdóttir[1] and Andrea Schalk*

[1] **BRICS** *** Dep. of Computer Science, Aalborg University, Denmark†
[2] Computer Laboratory, University of Cambridge, Cambridge, United Kingdom‡2

Abstract. A domain theoretic denotational model is given for a simple sublanguage of CCS extended with divergence operator. The model is derived as an abstraction on a suitable notion of normal forms for labelled transition systems. It is shown to be fully abstract with respect to observational precongruence.

1 Introduction

In describing the semantics of communicating processes the notion of bisimulation, [Par81, Mil83], has became standard in the literature. In this setting two processes are considered to be behaviourally equivalent if they can simulate each other's behaviour. It is standard practice to distinguish between strong bisimulation, where the silent τ-moves are considered visible, and weak bisimulation, which abstracts away from them. Weak bisimulation equivalence turns out not to be a congruence with respect to some of the standard operators found in process algebras, e.g. the choice operator of CCS, and therefore the notion of weak bisimulation congruence, often called observational congruence, has been introduced.

One of the main characteristics of weak bisimulation equivalence, and of the associated congruence, is the fact that it allows for the abstraction from divergence, i.e. infinite sequences of internal computations, in process behaviours. Semantic theories for processes based on the bisimulation idea which take divergence into account have also been considered in the literature, see, e.g., [Wal90, AH92]. In those studies, bisimulation equivalence is extended to a bisimulation preorder, usually referred to as *prebisimulation*. Intuitively if a process p is smaller than a process q with respect to the bisimulation preorder, then the two processes are bisimilar, but p may diverge more often than q.

For a further understanding and justification of the idea of bisimulation, many researchers have presented more abstract formal descriptions for it, using different theoretical tools to analyse in depth the nature of the concept. Examples

* Partly supported by an EC Human Capital and Mobility Grant

*** Basic Research in Computer Science, Centre of the Danish National Research Foundation.

† `email:` annai@iesd.auc.dk

‡ `email:` Andrea.Schalk@cl.cam.ac.uk

of such alternative descriptions are characterizations of bisimulation by means of modal logics, [Sti87, Mil89], sound and complete axiomatizations, [Hen81, Wal90, AH92], and fully abstract denotational models, [Hen81, Abr91, AH92]. The denotational models are usually given in terms of Σ-domains, where Σ is a signature, i.e. a set of syntactic operators. Typically this is the set of operators which are allowed in the language one wants to model. A Σ-domain is an ω-algebraic cpo endowed with a continuous Σ-algebra structure, i.e. a collection of continuous functions, one for each syntactic operator in Σ.

The existence of a fully abstract model in terms of a Σ-domain has the consequence that the behavioural preorder one is trying to model has to be finitary. Intuitively this means that the behavioural preorder is completely induced by finite observations of process behaviours. This is, in general, not the case for bisimulation as shown in, e.g., [Abr91]. For this reason, studies on mathematical models for such relations usually focus on providing denotational models for *finitary* versions of the bisimulation preorders [Hen81, Abr91, AH92].

There is a natural connection between sound and complete axiomatizations of behavioural preorders and fully abstract denotational models. Denotational models are often given in terms of initial Σ-domains satisfying a set of inequations [Hen88a]. In this kind of models the interpretation of a term is simply the set of terms which can be proved equal to it by the proof system. This type of denotational models is usually referred to as term models in the literature. Examples of such models may be found in, e.g., [Hen81], where the author defines a fully abstract term model for strong prebisimulation on a simple extension of SCCS, and in [AH92], where the authors give a fully abstract term model for observational congruence over an extension of the standard CCS.

Term models have been criticized for not giving much more insight into the semantics than the proof system already does. It is true that the existence of a fully abstract model for a behavioural preorder in terms of an algebraic cpo does imply that the behavioural preorder must be finitary, but usually this property has to be proven first anyway to prove the full abstractness of such a model. On the other hand by giving a syntax free representation of the term model we may gain some insight into the properties of the semantics we want to model. One way of obtaining such a syntax free representation, which is fully abstract with respect to a finitary behavioural preorder, is to investigate the preorder on the process graphs, that define the operational semantics. By investigating this preorder for finite processes we may gain enough information to be able to predict the behaviour for infinite processes. This is typically done by introducing some notion of semantic normal form for process graphs which contains enough information about the behavioural preorder. This kind of semantic normal forms will then induce a poset which coincides with the kernel of the behavioural preorder for finite processes. If all processes in the language can be turned into syntactic normal forms (i.e process terms whose process graph is in semantic normal form) in a sound way, i.e. preserving the behavioural semantics, it is sometimes possible to obtain a fully abstract model by taking the unique algebraic cpo which has the poset derived from semantic normal forms as representation of its compact elements. A similar approach occurs in Hennessy's model for testing equivalence based on finite acceptance trees which basically is a syntax free representation of the syntactic normal forms he defines, [Hen85, Hen88b].

Yet another, and maybe the most mathematically elegant, way of defining a denotational model is to define it as the initial solution to a recursive domain equation. The normal forms may now occur in the description of the compact elements derived from this definition. In [Abr91] Abramsky defines a fully abstract model for the finitary part of strong prebisimulation for SCCS. The domain theoretic constructions he uses are a variant of the Plotkin power construction and a notion of infinite sum. The poset of compact elements of the model may, roughly speaking, be represented as finite convex closed sets of finite synchronization trees ordered by the strong bisimulation preorder.

To give a short summary of the existing denotational models for bisimulation we have: For strong ω-prebisimulation on SCCS a fully abstract term model is given in [Hen81] and a fully abstract syntax free model in [Abr91]. For the ω-version of observational precongruence a fully abstract term model is given in [AH92] for an extension of CCS.

In this paper we will contribute to the investigation in a denotational setting of bisimulation preorders by defining a syntax free model for the ω-observational precongruence. Our aim is therefore to define a Σ-domain which is initial in the class of Σ-domains satisfying the set of equation that characterizes the ω-observational congruence and which does not mention terms or equations. Our approach is based on the idea of normal forms and ideal closure as described above. Thus we introduce semantic normal forms which are simply the process graphs derived from the syntactic normal forms introduced by Walker in [Wal90] ordered by *strong* bisimulation preorder. These normal forms may be represented as restricted form for finite synchronization trees ordered by the Egli-Milner preorder and can therefore be compared to Abramsky's model in [Abr91]. Our hope was to, instead of using equivalence classes as the elements of our model, to represent the equivalence classes by some canonical elements, i.e. the normal forms. Unfortunately our approach did not work quite as well as we had hoped. All the equations turn out to be sound in this model apart from the equation

$$\mu.\tau.x = \mu.x \tag{1}$$

which turns out to be difficult to model. Therefore our model is structured on two levels: on the first level we define the normal synchronization trees as described above whereas we obtain the second level by factoring out equation (1). We show the full abstractness of our model with respect to ω-observational precongruence.

We will focus on much simpler language than the one studied in [AH92] as the signature we consider only consists of action prefixing, the choice operator and the operators *nil* for the convergent inactive operator and Ω for the inactive divergent operator. Thus we only consider a language describing trees, finite or infinite which leaves are either *nil* or Ω. All the aspects we are interested in investigating are captured by this simple language. (Most of the results we obtain may be easily extended to full CCS or similar languages with divergence added to them.) In our study we make an extensive use of properties already proven in the literature, e.g. in the definition of the existing models for bisimulation preorders. Thus we may assume the soundness and completeness of the proof system used to define the term model in [AH92] which of course is simplified and modified according to the different language. In the same reference it is also proved that the ω-observational precongruence is finitary. From [Wal90] we borrow a suitable notion of normal forms and a corresponding normalization theorem. Furthermore we also get from that reference an alternative characterization of the observational precongruence which turns out to be useful in our studies.

The structure of the paper is as follows: In Section 2 we give a short review of labelled transition systems with divergence and the diverse notions of prebisimulation and observational precongruence; in the same section we introduce the notion of normal forms and give an alternative characterization of the observational precongruence on these. In Section 3 we define the language $Trees$ and give a short summary of existing results for this: a sound and complete axiomatization of the observational preorder and a normalization theorem for finite trees. In Section 4 we give a short overview over the domain theory we need whereas Section 5 is devoted to the definition of our domain and a proof of a full abstractness with respect to observational precongruence. We finish the paper by giving some concluding remarks.

2 Labelled Transition Systems with Divergence

The operational semantics of the languages considered in this paper will be given in terms of a variation on the notion of labelled transition systems [Kel76] that takes divergence information into account. We refer the interested readers to, e.g., [Hen81, Mil81, Wal90] for motivation and more information on (variations of) this semantic model for reactive systems.

Definition 1. [Labelled Transition Systems with Divergence] A *labelled transition system with divergence (lts)* is a quadruple $(\mathsf{P}, \mathsf{Lab}, \rightarrow, \uparrow)$, where:

- P is a set of *processes* or *states*, ranged over by s, s', s_i;
- Lab is a set of *labels*, ranged over by ℓ;
- $\rightarrow \subseteq \mathsf{P} \times \mathsf{Lab} \times \mathsf{P}$ is a *transition relation*. As usual, we shall use the more suggestive notation $s \stackrel{\ell}{\rightarrow} s'$ in lieu of $(s, \ell, s') \in \rightarrow$;
- $\uparrow \subseteq \mathsf{P}$ is a *divergence predicate*, notation $s \uparrow$.

We let Λ range over all lts's. A *process graph* is a pair of the form (s_0, Λ) where $s_0 \in \mathsf{P}$ is the *initial state* and P is the set of processes in Λ.

We write $s \downarrow$, read "s converges globally", iff it is not the case that $s \uparrow$. The *lts* $\Lambda = (\mathsf{P}, \mathsf{Lab}, \rightarrow, \uparrow)$ is said to be a *finite state* lts if P is finite and *acyclic* if its derived graph does not contain cycles; it is said to be *finite* if it is a finitely branching finite state lts. A *finite tree lts* is defined in the obvious way. We note here that each acyclic finite lts may be turned into a finite tree lts by making one copy of each state for each incoming arc where the outgoing arcs and their descendants are the same as from the original state. We note that the resulting lts is not graph isomorphic to the original one as in general it has more states. On the other hand it is strongly bisimilar to the original one in the sense to be defined later in this section. In our semantic theory the operational semantics for a process s, is given by a process graph with the process as the initial state. Intuitively, a process graph is an lts with a pointer to the initial state. If the underlying lts Λ is fixed we write s instead of the process graph (s, Λ). A finite tree lts Λ, has a canonical initial state, namely the root of the tree, $root(\Lambda)$. Therefore, if Λ is a finite tree lts, we often refer to the process graph $(root(\Lambda), \Lambda)$ as Λ. In this study we will follow this practice without further explanations.

We define the following operators on process graphs:

1. Action prefixing:
   $l : (s, (P, \longrightarrow, \uparrow)) = (l : s, (\mathsf{P} \cup \{l : s\}, \mathsf{Lab} \cup \{l\}, \longrightarrow \cup \{(l : s, l, s)\}, \uparrow))$ where $l : s \notin \mathsf{P}$ is a new state.
2. Finite sum:
   $\sum_{i \le N}(s_i, (\mathsf{P}_i, \mathsf{Lab}_i, \longrightarrow_i, \uparrow_i)) = (s', (\mathsf{P}', \mathsf{Lab}', \longrightarrow', \uparrow'))$ where
   (a) $N > 0$,
   (b) $s' = \sum_{i \le N} s_i \notin \bigcup_{i \le N} \mathsf{P}_i$ is a new state,
   (c) $\mathsf{P}' = \{s'\} \cup \bigcup_{i \le N}(\mathsf{P}_i \setminus \{s_i\})$,
   (d) $\mathsf{Lab}' = \bigcup_{i \le N} \mathsf{Lab}_i$,
   (e) $\longrightarrow' = \bigcup_{i \le N}(\longrightarrow_i \setminus \{(s_i, l, s'_i) | \ (s_i, l, s'_i) \in \longrightarrow_i\}$
   $\cup \{(s', l, s'_i) | \ (s_i, l, s'_i) \in \longrightarrow_i\})$,
   (f) $\uparrow' = \bigcup_{i \le N} \uparrow_i \cup \{s' | \ \exists j . s_j \in \uparrow_j\}$.

We use the infix notation $\oplus$ for the sum over two lts's. We assume that $\mu : \_$ has priority over $\oplus$. The finite tree lts's may be represented as the set of finite synchronization trees over a set of labels Lab, denoted by $\mathsf{ST}(\mathsf{Lab})$. These are the sets generated by the following inductive definition:

1. $\emptyset, \{\perp\} \in \mathsf{ST}(\mathsf{Lab})$,
2. $\ell_i \in \mathsf{Lab}, t_i \in \mathsf{ST}(\mathsf{Lab})$ for $i \le N$ implies $\{\langle \ell_i, t_i \rangle \mid i \le N\}[\cup\{\perp\}] \in \mathsf{ST}(\mathsf{Lab})$,

where the notation $[\cup\{\perp\}]$ means optional inclusion of $\perp$. The divergence predicate and the transition relation are defined as follows:

- $t \uparrow$ iff $\perp$ is in t, and
- $t \xrightarrow{\ell_i} t_i$ iff $\langle \ell_i, t_i \rangle$ is in t.

We let t range over $ST(\mathsf{Lab})$.

We note that the process graph operators also apply to $ST(\mathsf{Lab})$; in this representation we have $\mu : t = \{\langle \mu, t \rangle\}$ and $\oplus = \cup$. Furthermore we often write μ instead of $\{\langle \mu, \emptyset \rangle\}$ or $\mu : \emptyset$. This will simplify our notation considerably later on. (The reader should keep in mind that in this case $t_1 \oplus t_2$ and $l : t$ are synchronization trees and therefore sets.)

The following norm on $ST(\mathsf{Lab})$ will be needed in this study.

The depth of a synchronization tree, $d : ST(\mathsf{Lab}) \longrightarrow Nat$ is defined by

1. $d(\emptyset) = d(\{\perp\}) = 0$
2. $d(\{\langle \mu, t\rangle\}) = 1 + d(t)$
3. $(\{\langle \mu_i, t_i\rangle | i \leq N\}[\oplus\{\perp\}]) = \max\{d(\{\langle \mu_i, t_i\rangle\} | i \leq N\})$

We extend the function d to $d : ST(\mathsf{Lab}) \times ST(\mathsf{Lab}) \longrightarrow Nat$ by $d(t_1, t_2) = d(t_1) + d(t_2)$.

In what remains of this section we let $\Lambda = (P, Lab, \rightarrow, \uparrow)$ be a fixed lts. Furthermore we let $Rel(P)$ denote the set of binary relations over P. The behavioural relations over processes that we shall study in this paper are those of *prebisimulation* [Mil81, Hen81, Wal90]. (also known as *partial bisimulation* [Abr91]).

Definition 2 Strong Prebisimulation. Define the functional $\mathcal{F}_s : Rel(P) \rightarrow Rel(P)$ (s stands for "strong") by:

Given $\mathcal{R} \in \mathsf{Rel(P)}$, $s_1 \mathcal{F}_s(\mathcal{R}) s_2$ iff, for each $l \in Act_\tau$,

1. If $s_1 \stackrel{l}{\longrightarrow} s_1'$ then, for some s_2', $s_2 \stackrel{l}{\longrightarrow} s_2'$ and $s_1' \mathcal{R} s_2'$.
2. If $s_1 \downarrow$ then
   (a) $s_2 \downarrow$ and
   (b) if $s_2 \stackrel{l}{\longrightarrow} s_2'$ then, for some s_1', $s_1 \stackrel{l}{\longrightarrow} s_1'$ and $s_1' \mathcal{R} s_2'$.

The strong prebisimulation preorder (over Λ), $\precsim_\Lambda$ is defined as the largest fixed-point for $\mathcal{F}_s$. If Λ is known from the context we write $\precsim$ instead of $\precsim_\Lambda$.

The relation $\precsim$ is a preorder over P and its kernel will be denoted by $\sim$, i.e., $\sim = \precsim \cap \precsim^{-1}$. Intuitively, $s_1 \precsim s_2$ if s_2's behaviour is at least as specified as that of s_1, and s_1 and s_2 can simulate each other when restricted to the part of their behaviour that is fully specified. A divergent state s with no outgoing transition is a least element with respect to $\precsim$ and intuitively corresponds to a process whose behaviour is totally unspecified — essentially an operational version of the bottom element $\perp$ in Scott's theory of domains [SS71, Plo81].

The preorder $\precsim$ (and other similar relations) is extended to process graphs by

$$(s_1, \Lambda_1) \precsim (s_2, \Lambda_2) \text{ if and only if } s_1 \precsim_{\Lambda_1 \uplus \Lambda_2} s_2,$$

where $\Lambda_1 \uplus \Lambda_2$ is the standard disjoint union of Λ_1 and Λ_2. Processes from different lts's are compared in this way where we usually write only $s_1 \precsim s_2$. In the sequel, this will be done without further comment. (We will often need to compare states in an lts with finite synchronization trees.)

In this study, we shall be interested in relating the notion of prebisimulation to a preorder on processes induced by a denotational semantics given in terms of an algebraic domain [Plo81]. As such preorders are completely determined by how they act on *finite processes*, we shall be interested in comparing them with the "finitely observable", or *finitary*, part of the bisimulation in the sense of, e.g., [Gue81, Hen81]. The following definition is from [Abr91].

Definition 3. Let $\mathcal{R} \in \mathsf{Rel(P)}$. The *finitary part of* $\mathcal{R}$, $\mathcal{R}^F$, is defined on any lts by

$$s \mathcal{R}^F s' \Leftrightarrow \forall t \in ST(Lab).\ t \mathcal{R} s \Rightarrow t \mathcal{R} s' \ .$$

An alternative method for using the functional $\mathcal{F}_s$ to obtain a behavioural preorder is to apply it inductively as follows:

- $\sqsubset\!\!\!\sim_0 = \mathsf{P} \times \mathsf{P}$,
- $\sqsubset\!\!\!\sim_{n+1} = \mathcal{F}_s(\sqsubset\!\!\!\sim_n)$,

and finally $\sqsubset\!\!\!\sim_\omega = \bigcap_{n \geq 0} \sqsubset\!\!\!\sim_n$. Intuitively, the preorder $\sqsubset\!\!\!\sim_\omega$ is obtained by restricting the prebisimulation relation to observations of finite depth. The preorders $\sqsubset\!\!\!\sim$, $\sqsubset\!\!\!\sim_\omega$ and $\sqsubset\!\!\!\sim^F$ are, in general, related thus:

$$\sqsubset\!\!\!\sim \;\subseteq\; \sqsubset\!\!\!\sim_\omega \;\subseteq\; \sqsubset\!\!\!\sim^F .$$

Moreover the inclusions are, in general, strict. The interested reader is referred to [Abr91] for a wealth of examples distinguishing these preorders, and a very deep analysis of their general relationships and properties. Here we simply state the following useful result, which is a simple consequence of [Abr91, Lem. 5.10]:

Lemma 4. *For every $t \in \mathsf{ST}(\mathsf{Lab})$, $s \in \mathsf{P}$, $t \sqsubset\!\!\!\sim s$ iff $t \sqsubset\!\!\!\sim_\omega s$.*

Next we define the weak version of the prebisimulation and the derived observational precongruence. Following the standard practice we assume that the set of labels Lab has the form $Act_\tau = Act \cup \{\tau\}$ where Act is a set of observable actions and $\tau \notin Act$ is an unobservable action. We let a range over Act and μ over Act_τ. We let $\stackrel{\mu}{\Longrightarrow}$ denote $(\stackrel{\tau}{\longrightarrow})^* \cdot \stackrel{\mu}{\longrightarrow} \cdot (\stackrel{\tau}{\longrightarrow})^*$. So $s_1 \stackrel{\mu}{\Longrightarrow} s_2$ means that s_1 may evolve to s_2 performing the action μ and possibly invisible actions. We will also use the relation $\stackrel{\varepsilon}{\Longrightarrow}$, defined as $(\stackrel{\tau}{\longrightarrow})^*$.

For any s, let $Sort(s) = \{\mu \in Act_\tau \mid \exists \sigma \in Act_\tau^*, s' \in P : s \stackrel{\sigma\mu}{\longrightarrow} s'\}$, where, for $\sigma \in Act_\tau^*$, $\stackrel{\sigma}{\longrightarrow}$ is defined in the natural way. In this study we only deal with lts's which are sort finite, that is where $Sort(s)$ is finite for each $s \in \mathsf{P}$. Some of our results will depend on this fact.

Processes that can perform an infinite sequence of τ-actions are weakly divergent, which brings us to a definition of a weak divergence predicate. Let $\Downarrow$ be the least predicate over P which satisfies

$$s_1 \downarrow \text{ and (for each } s_2,\ s_1 \stackrel{\tau}{\longrightarrow} s_2 \text{ then } s_2 \Downarrow) \text{ imply } s_1 \Downarrow .$$

$s \Uparrow$ means that $s \Downarrow$ is not the case. In the semantic preorder to be defined we will use versions of $\Downarrow$ which are parameterized by actions:

$$s_1 \Downarrow \mu \text{ if } s_1 \Downarrow \text{ and , for each } s_2,\ s_1 \stackrel{\mu}{\Longrightarrow} s_2 \text{ implies } s_2 \Downarrow .$$

We use the standard notation $\hat{\mu}$ where $\hat{\tau}$ stands for ε and $\hat{a}$ stands for a. The following definition is taken directly from [Wal90].

Definition 5. Given $\mathcal{R} \in \mathsf{Rel}(\mathsf{P})$, $s_1 \mathcal{F}_w(\mathcal{R}) s_2$ (w for "weak") iff, for each $\mu \in Act_\tau$,

1. if $s_1 \stackrel{\mu}{\longrightarrow} s_1'$ then, for some s_2', $s_2 \stackrel{\hat{\mu}}{\Longrightarrow} s_2'$ and $s_1' \mathcal{R} s_2'$
2. if $s_1 \Downarrow \mu$ then

(a) $s_2 \Downarrow \mu$
(b) if $s_2 \xrightarrow{\mu} s_2'$ then, for some s_1', $s_1 \xRightarrow{\hat{\mu}} s_1'$ and $s_1' \mathcal{R} s_2'$

The weak bisimulation preorder $\sqsubseteq_{\approx}$ is defined as the largest fixed-point for $\mathcal{F}_w$. The weak ω-bisimulation preorder $\sqsubseteq_{\approx\omega}$ is defined by

- $\sqsubseteq_{\approx 0} = \mathsf{P} \times \mathsf{P}$ (the top element in the lattice $(\mathsf{Rel}(\mathsf{P}), \subseteq)$)
- $\sqsubseteq_{\approx n+1} = \mathcal{F}_w(\sqsubseteq_{\approx n})$

and finally $\sqsubseteq_{\approx\omega} = \bigcap_{n\geq 0} \sqsubseteq_{\approx n}$.

Obviously $\sqsubseteq_{\approx} = \sqsubseteq_{\approx\omega}$ when restricted to finite acyclic process graphs. The following result is proved in [AH92].

Lemma 6. *For all $t \in \mathsf{ST}(Act_\tau)$ and $s \in \mathsf{P}$, $t \sqsubseteq_{\approx\omega} s$ iff $t \sqsubseteq_{\approx} s$.*

The set Act_τ is assumed to be fixed throughout the paper and from now on we write ST instead of $\mathsf{ST}(Act_\tau)$.

As it is well know from the literature, [Mil83, Mil89, Wal90], the preorder $\sqsubseteq_{\approx}$ is not a precongruence with respect to some of the standard operators, e.g. the choice operator $+$ of CCS. In terms of process graphs this is also the case. Thus the notion of observational precongruence is introduced. This will be done in the following:

For any $\mathcal{R} \in \mathsf{Rel}(\mathsf{P})$ we define the new relation $\mathcal{R}^c$ by:

$$s_1 \mathcal{R}^c s_2 \text{ if, for every context } \mathcal{C}[\cdot],\ \mathcal{C}[s_1] \mathcal{R} \mathcal{C}[s_2].$$

where a context for process graphs has the obvious meaning. Then $\mathcal{R}$ is said to be closed with respect to contexts if $\mathcal{R} = \mathcal{R}^c$. The observational precongruence is defined as $\sqsubseteq_{\approx}^c$ and may be described as the least precongruence contained in weak bisimulation preorder. In [Wal90] Walker gives an operational characterization of $\sqsubseteq_{\approx}^c$. In order to obtain this he defines the operator $\_^*$ on $\mathsf{Rel}(\mathsf{P})$ as follows:

Definition 7. For all $\mathcal{R} \in \mathsf{Rel}(\mathsf{P})$ we let $s_1 \mathcal{R}^* s_2$ iff

1. if $s_1 \xrightarrow{a} s_1'$ then, for some s_2', $s_2 \xRightarrow{a} s_2'$ and $s_1' \mathcal{R} s_2'$,
2. if $s_1 \xrightarrow{\tau} s_1'$ then
   (a) if $s_1' \Downarrow$ then there exists s_2' such that $s_2 \xRightarrow{\tau} s_2'$ and $s_1' \mathcal{R} s_2'$,
   (b) if $s_1' \Uparrow$ then there exists s_2' such that $s_2 \xRightarrow{\varepsilon} s_2'$ and $s_1' \mathcal{R} s_2'$,
3. if $s_1 \Downarrow \mu$ then
   (a) $s_2 \Downarrow \mu$,
   (b) if $s_2 \xrightarrow{\mu} s_2'$ then, for some s_1', $s_1 \xRightarrow{\mu} s_1'$ and $s_1' \mathcal{R} s_2'$.

The following lemma is proved in [Wal90].

Lemma 8. $\sqsubseteq_{\approx}^c = \sqsubseteq_{\approx}^*$ *and* $\sqsubseteq_{\approx\omega}^c = \sqsubseteq_{\approx\omega}^*$.

The following definition of normal forms for synchronization trees is also borrowed from [Wal90] with an obvious adaption to process graphs.

Definition 9. (Normal Forms) An element $n = \{\langle\mu_1, n_1\rangle, \ldots, \langle\mu_l, n_l\rangle\}[\cup\{\bot\}] \in \mathsf{ST}$ (where $\cup\{\bot\}$ is optional) is a normal form if

1. n_i is a normal form for $i \leq l$,
2. if $\mu_i = \tau$ then $n_i \Downarrow$,
3. if $n \Downarrow$ and $n \Uparrow a$ then $\langle a, \{\bot\}\rangle \in n$,
4. if $n \stackrel{\mu}{\Longrightarrow} n'$ then $\langle\mu, n'\rangle \in n$.

Note that if n is a normal form then $n \Uparrow$ iff $n \uparrow$ iff $\bot \in n$. This property will play an important role in our investigation of the preorder $\sqsubseteq_{\approx}^{*}$ on normal forms. Now we will give a simple characterization of the normal forms as a subset of ST. For this purpose we introduce the following operators on ST:

Definition 10. We define μ_{NST} by:

$$\tau_{\mathsf{NST}}.t = t \cup \{\langle\tau, t\rangle | \bot \notin t\}$$
$$a_{\mathsf{NST}}.t = \{\langle a, t\rangle\} \cup \{\langle a, t'\rangle | \langle\tau, t'\rangle \in t\} \cup \{\langle a, \{\bot\}\rangle | \bot \in t\}$$

Now we define the subset NST of ST as follows:

Definition 11. We define the set NST as the smallest set which satisfies:

1. $\{\bot\}, \emptyset \in \mathsf{NST}$.
2. $n \in \mathsf{NST}$ and $\mu \in Act_\tau$ implies $\mu_{\mathsf{NST}}.n \in \mathsf{NST}$.
3. $n_1, n_2 \in \mathsf{NST}$ implies $n_1 \oplus n_2 \in \mathsf{NST}$.

We have the following lemma:

Lemma 12.

1. *$\mu_{\mathsf{NST}}.\_$ and $\oplus$ preserve $\sqsubseteq_{\approx}$.*
2. *If $n = \sum_{i \leq N} \mu_i : n_i[\oplus\{\bot\}]$ is a normal form then $n = \sum_{i \leq N} \mu_{i\mathsf{NST}}.n_i[\oplus\{\bot\}]$.*
3. *The set NST is exactly the subset of normal forms of ST.*

Proof.

1. Straight forward and is left to the reader.
2. Let $m = \sum_{i \leq N} \mu_{i\mathsf{NST}}.n_i[\oplus\{\bot\}]$. We will prove that $n = m$ where "=" is set equality. The inclusion $n \subseteq m$ is obvious. To prove that $m \subseteq n$, assume that $\langle\mu, m'\rangle \in m$. We will prove that $\langle\mu, m'\rangle \in n$. We know that $\langle\mu, m'\rangle \in \mu_{i\mathsf{NST}}.n_i$ for some i. If $\langle\mu, m'\rangle = \langle\mu_i, n_i\rangle$ we are done so assume this is not the case. We have the following two cases:
   $\mu_i = \tau$: Then either $\tau_{\mathsf{NST}}.n = n$ or $\tau_{\mathsf{NST}}.n = \{\langle\tau, n\rangle\} \cup n$. As by assumption $\langle\mu, m'\rangle \neq \langle\tau, n\rangle$ this implies $\langle\mu, m'\rangle \in n$.
   $\mu_i = a \in Act$: Then

$$\mu_{i\mathsf{NST}}.n_i = a : n_i \cup \{\langle a, n_i'\rangle | \langle\tau, n_i'\rangle \in n_i\} \cup [\{\langle a, \{\bot\}\rangle | \bot \in n_i\}].$$

   Now either $\langle\mu, m'\rangle = \langle a, n_i'\rangle$ for some n_i' where $\langle\tau, n_i'\rangle \in n_i$ or $\langle\mu, m'\rangle = \langle a, \{\bot\}\rangle$ where $\bot \in n_i$. In both cases, by definition of normal forms, $\langle\mu, m'\rangle \in n$.

3. Let ST_N denote the subset of normal forms in ST. We will prove that $ST_N = NST$. By a simple induction on the definition of NST we may prove that $NST \subseteq ST_N$. To prove that $ST_N \subseteq NST$, let
$$n = \sum_i \mu_i : n_i[\oplus\{\bot\}] \in ST_N.$$
By part 2. of the lemma
$$n = \sum_i \mu_{i\,NST}.n_i[\oplus\{\bot\}],$$
which in turn implies that $n \in NST$.

In the following we define a finer version of a preorder originally defined in [Wal90]. (In the set of normal forms these two definitions coincide.) It gives a simplified characterization of the preorders $\sqsubseteq_{\approx}$ and $\sqsubseteq_{\approx}^{*}$ on NST.

Definition 13.

1. We define $\mathcal{F}^g_w : \mathsf{Rel}(\mathsf{P}) \longrightarrow \mathsf{Rel}(\mathsf{P})$ (where g stands for "global convergence") by: Given $\mathcal{R} \in \mathsf{Rel}(\mathsf{P})$, $s_1\mathcal{F}^g_w(\mathcal{R})s_2$ iff, for each $a \in Act$,
   (a) if $s_1 \xrightarrow{a} s_1'$ then, for some s_2', $s_2 \xrightarrow{a} s_2'$ and $s_1'\mathcal{R}s_2'$,
   (b) if $s_1 \xrightarrow{\tau} s_1'$ then, for some s_2', $s_2 \xRightarrow{\varepsilon} s_2'$ and $s_1'\mathcal{R}s_2'$,
   (c) if $s_1 \downarrow$ then the following holds:
      i. $s_2 \downarrow$,
      ii. if $s_2 \xrightarrow{a} s_2'$ then, for some s_1', $s_1 \xrightarrow{a} s_1'$ and $s_1'\mathcal{R}s_2'$,
      iii. if $s_2 \xrightarrow{\tau} s_2'$ then, for some s_1', $s_1 \xRightarrow{\varepsilon} s_1'$ and $s_1'\mathcal{R}s_2'$.

   We define $\sqsubseteq_{\approx g}$ to be the largest fixed point of $\mathcal{F}^g_\omega$.
2. We define the preorder $\sqsubseteq_{\approx g}^{\diamond}$ by: $s_1 \sqsubseteq_{\approx g}^{\diamond} s_2$ iff, for each $\mu \in Act_\tau$,
   (a) if $s_1 \xrightarrow{\mu} s_1'$ then, for some s_2', $s_2 \xrightarrow{\mu} s_2'$ and $s_1' \sqsubseteq_{\approx g} s_2'$,
   (b) if $s_1 \downarrow$ then
      i. $s_2 \downarrow$ and
      ii. if $s_2 \xrightarrow{\mu} s_2'$ then, for some s_1', $s_1 \xrightarrow{\mu} s_1'$ and $s_1' \sqsubseteq_{\approx g} s_2'$.

In [Wal90] the author shows that in general $\sqsubseteq_{\approx g}$ is strictly finer than the weak bisimulation preorder $\sqsubseteq_{\approx}$. However it turns out that for normal forms these two preorders and their derived preorders, $\sqsubseteq_{\approx}^{*}$ and $\sqsubseteq_{\approx g}^{\diamond}$, coincide. This is the content of the following theorem.

Theorem 14. *For all $n_1, n_2 \in \mathsf{NST}$, $n_1 \sqsubseteq_{\approx} n_2$ iff $n_1 \sqsubseteq_{\approx g} n_2$ and $n_1 \sqsubseteq_{\approx}^{*} n_2$ iff $n_1 \sqsubseteq_{\approx g}^{\diamond} n_2$.*

Proof. See Appendix A.

We observe that the characterization $\sqsubseteq_{\approx s}^{\diamond}$ of the preorder $\sqsubseteq_{\approx}^{*}$ on normal forms looks very much like the definition for the strong prebisimulation preorder $\sqsubseteq_{\sim}$. The only difference is that on lower levels a τ transition may be matched by an empty transition. The following example shows that with the definition of normal forms we have chosen the preorders $\sqsubseteq_{\approx}^{*}$ and $\sqsubseteq_{\sim}$ do indeed not coincide.

Example 1. Let $n_1 = \tau : (\tau : a \oplus a \oplus a : \{\bot\}) \oplus \tau : a \oplus a \oplus a : \{\bot\}$ and $n_2 = \tau : a \oplus a$. Then $n_1 \sqsubseteq^*_{\approx} n_2$ but $n_1 \not\sqsubseteq_{\approx} n_2$. The reason for this is that the left hand side can perform a sequence of two τs to start with while the left hand side only can perform a sequence of τ-transitions of length one. On the other hand if we add $\tau : (\tau : a \oplus a)$ (adding only $\tau : \tau : a$ would not preserve the normal form property) to the right hand side the τ-depth is balanced and we get that $n_1 \sqsubseteq_{\approx} n_2 \oplus \tau : (\tau : a \oplus a)$. Furthermore $n_2 \approx^* n_2 \oplus \tau : (\tau : a \oplus a)$ (where $\approx^*$ is the kernel of $\sqsubseteq^*_{\approx}$).

The example above illustrates that the preorders $\sqsubseteq^*_{\approx}$ and $\sqsubseteq_{\approx}$ do not coincide on NST. But at the same time it also suggests that if $n \sqsubseteq^*_{\approx} m$ then by performing a simple balancing operation on n and m, which is sound with respect to $\approx^*$ we may get a pair of normal forms, n' and m', where $n' \sqsubseteq_{\approx} m'$. In our attempt to give a simple characterization of the preorder $\sqsubseteq^*_{\approx}$ on normal forms this would be a useful result. In the next section we will therefore formalize this informal statement and prove that it holds.

2.1 The Characterization Theorem for Observational Precongruence

In the proof for the characterization result for $\sqsubseteq^*_{\approx}$ on NST outlined above we suggested a transformations on normal forms which is sound with respect to $\sqsubseteq^*_{\approx}$. To formalize this idea we introduce a notion of equivalence on NST meaning that two elements are equivalent if they can be transformed into the same terms applying the balancing operation described above. As we need to be able to apply the transformation mentioned above recursively on the structure of the normal form we want the equivalence to be a congruence with respect to the graph operators. On the other hand the operator $\mu : \_$ does not preserve normal forms whereas the operator μ_{NST} does. Also the operator $\oplus$ preserves normal forms. Furthermore we know from Lemma 12 that for normal forms

$$\sum_{i \leq N} \mu_i : n_i[\oplus\{\bot\}] = \sum_{i \leq N} \mu_{i\,NST}.n_i[\oplus\{\bot\}].$$

Keeping this in mind we only require the equivalence to be a congruence with respect to the operators $\mu_{NST}.\_$ and $\oplus$. To define the equivalence suggested above we introduce the equation

$$\tau 1 \qquad\qquad \mu.\tau.x = \mu. \tag{2}$$

This equation is interpreted over NST with respect to the operators $\mu_{NST}.\_$ above. Now we let $=_{\tau 1}$ denote the least congruence over NST with respect to $\mu_{NST}.\ \_$ and $\oplus$ generated by the equation $\tau 1$ in (2). It is easy to see that this equation is sound with respect to $\approx^*$ on NST, i.e. that $=_{\tau 1} \subseteq \approx^*$ on NST.

Furthermore we need the following general theorem which is a slight modification of a similar theorem proved in [AH92].

Theorem 15 Hennessy's Theorem. *For all $t_1, t_2 \in \mathsf{ST}$ the following holds:*

$$t_1 \sqsubseteq_{\approx} t_2 \textit{ iff } t_1 \sqsubseteq^*_{\approx} t_2 \textit{ or } t_1 \sqsubseteq^*_{\approx} \tau_{\mathsf{NST}}.t_2 \textit{ or } \tau_{\mathsf{NST}}.t_1 \sqsubseteq^*_{\approx} t_2.$$

The Characterization Theorem is now stated as follows:

Theorem 16 The Characterization Theorem. *Let $n, m \in \mathsf{NST}$. Then $n \sqsubseteq_{\approx}^{*} m$ if and only if there exist some $n', m' \in \mathsf{NST}$ such that $n' \sqsubseteq_{\sim} m'$, $n =_{\mathcal{T}1} n'$ and $m =_{\mathcal{T}1} m'$.*

Proof. The "if" part follows immediately. To prove the "only" part assume $n \sqsubseteq_{\approx}^{*} m$. We proceed by induction on the combined depth of n and m, $d(n, m)$.

$d(n, m) = 0$: The only possible combinations are the following:
1. $n = m = \emptyset$,
2. $n = \{\perp\}$ and $m = \emptyset$, and
3. $n = m = \{\perp\}$.

All three cases are obvious.

$d(n, m) = k + 1$: Assume

$$n = \sum_{i \le N} \mu_{i\,\mathsf{NST}}.n_i[\oplus\{\perp\}] \text{ and } m = \sum_{i \le M} \gamma_{j\,\mathsf{NST}}.m_j[\oplus\{\perp\}].$$

By Lemma 12 we get

$$n = \sum_{i \le N} \mu_i : n_i[\oplus\{\perp\}] \text{ and } m = \sum_{i \le M} \gamma_j : m_j[\oplus\{\perp\}].$$

First let us assume that $\perp \notin n$ and therefore $\perp \notin m$. Now we recall that if $n \xrightarrow{\mu} n'$ then $m \xrightarrow{\mu} m'$ for some m' where $n' \sqsubseteq_{\approx} m'$ and vice versa. We may therefore assume that $N = M$ and that $\mu_i = \gamma_i$ and $n_i \sqsubseteq_{\approx} m_i$ for $i \le N$. (We may have to rearrange the summands and/or duplicate some of them as well to obtain this.) By Hennessy's Theorem 15, for each i one of the following holds:

Case 1: $n_i \sqsubseteq_{\approx}^{*} m_i$: By induction there are $n_i', m_i' \in \mathit{NST}$ such that $n_i =_{\mathcal{T}1} n_i'$, $m_i =_{\mathcal{T}1} m_i'$ and $n_i' \sqsubseteq_{\sim} m_i'$. Furthermore $\mu_{i\,\mathsf{NST}}.n_i =_{\mathcal{T}1} \mu_{i\,\mathsf{NST}}.n_i'$ and $\mu_{i\,\mathsf{NST}}.m_i =_{\mathcal{T}1} \mu_{i\,\mathsf{NST}}.m_i'$.

Case 2: $n_i \sqsubseteq_{\approx}^{*} \tau_{\mathsf{NST}}.m_i$: By induction there are $n_i', m_i' \in \mathit{NST}$ such that $n_i =_{\mathcal{T}1} n_i'$, $\tau_{\mathsf{NST}}.m_i =_{\mathcal{T}1} m_i'$ and $n_i' \sqsubseteq_{\sim} m_i'$. Furthermore $\mu_{i\,\mathsf{NST}}.n_i =_{\mathcal{T}1} \mu_{i\,\mathsf{NST}}.n_i'$ and $\mu_{i\,\mathsf{NST}}.m_i =_{\mathcal{T}1} \mu_{i\,\mathsf{NST}}.\tau_{\mathsf{NST}}.m_i =_{\mathcal{T}1} \mu_{i\,\mathsf{NST}}.m_i'$.

Case 3: $\tau_{\mathsf{NST}}.n_i \sqsubseteq_{\approx}^{*} m_i$: By induction there are $n_i', m_i' \in \mathit{NST}$ such that $\tau_{\mathsf{NST}}.n_i =_{\mathcal{T}1} n_i'$, $m_i =_{\mathcal{T}1} m_i'$ and $n_i' \sqsubseteq_{\sim} m_i'$. Furthermore $\mu_{i\,\mathsf{NST}} n_i =_{\mathcal{T}1} \mu_{i\,\mathsf{NST}}.\tau_{\mathsf{NST}}.n_i =_{\mathcal{T}1} \mu_{i\,\mathsf{NST}}.n_i'$ and $\mu_{i\,\mathsf{NST}}.m_i =_{\mathcal{T}1} \mu_{i\,\mathsf{NST}}.m_i'$.

We let

$$n' = \sum_{i \le N} \mu_{i\,\mathsf{NST}}.n_i'$$

and

$$m' = \sum_{i \le N} \mu_{i\,\mathsf{NST}}.m_i'$$

which both are normal forms. Obviously $n =_{\mathcal{T}1} n'$ and $m =_{\mathcal{T}1} m'$. Furthermore by Lemma 12,

$$n' = \sum_{i \leq N} \mu_i : n'_i$$

and

$$m' = \sum_{i \leq N} \mu_i : m'_i.$$

It is now easy to see that $n' \sqsubseteq_{\approx} m'$.

Next assume that $\perp \in n$. The case when it is the only element is obvious, so assume this is not the case. Now we recall that if $n \xrightarrow{\mu} n'$ then there is an m' such that $m \xrightarrow{\mu} m'$ and $n' \sqsubseteq_{\approx} m'$. By a similar reasoning as in the previous case we may now assume that

$$n = \sum_{i \leq N} \mu_i : n_i \oplus \{\perp\} \text{ and } m = \sum_{i \leq N} \mu_i : m_i \oplus m'$$

where $n_i \sqsubseteq_{\approx} m_i$ for $i \leq N$. We may also assume that $\sum_{i \leq N} \mu_i : m_i$ and m' are normal forms. Now the proof may proceed as in the previous case but now using the fact that $\{\perp\} \sqsubseteq_{\approx} m'$.

3 The Language

In this section we will give a short survey of the theory of observational precongruence for a simple sublanguage of CCS extended with the divergence operator. The language $Trees$ is a language that denotes trees, finite and infinite, and only contains the operators nil, $+$ and $\mu.\_$ which all have the standard meaning [Mil80], plus the nullary operator Ω, which stands for the inactive divergent process [Hen81, Wal90]. Infinite processes are given in the standard way by means of the construction $\texttt{rec}x.\_$ where x is a process variable.

Definition 17. Let $\texttt{Var}$ be a countable set of process variables, ranged over by $x, y \ldots$ and Act_τ have the same meaning as in the previous section, ranged over by μ. The syntax of the language $TreeTerms$ is defined by

$$u ::= nil \mid \Omega \mid \mu.u \mid u + u \mid x \mid \texttt{rec}x.u.$$

We let $Trees$ denote the set of closed terms in $TreeTerms$ and $FinTrees$ the set of recursion free elements of $Trees$. We let u range over $TreeTerms$, p, q over $Trees$ and d over $FinTrees$.

The operational semantics in terms of a transition relation and a convergence (and divergence) predicate is also defined in the standard way (see e.g. [Hen81, Wal90, AH92]).

Definition 18.

1. Let $\downarrow$ be the smallest subset of $Trees$ which satisfies
   (a) $nil\downarrow, \mu.p\downarrow$,
   (b) $p\downarrow$ and $q\downarrow$ implies $(p+q)\downarrow$,
   (c) $t[\texttt{rec}x.t/x]\downarrow$ implies $\texttt{rec}x.t\downarrow$.
   We say that $p\uparrow$ iff $p\downarrow$ is not true.
2. For each $\mu \in Act_\tau$, let $\xrightarrow{\mu}$ be the least binary relation on $Trees$ which satisfies the following axioms and rules:
   (a) $\mu.p \xrightarrow{\mu} p$,
   (b) $p \xrightarrow{\mu} p'$ implies $p+q \xrightarrow{\mu} p'$ and $q+p \xrightarrow{\mu} p'$,
   (c) $t[\texttt{rec}x.t/x] \xrightarrow{\mu} p'$ implies $\texttt{rec}x.t \xrightarrow{\mu} p'$.

This definition generates an lts, $\Lambda_{Tree} = (Trees, Act_\tau, \longrightarrow, \uparrow)$ which obviously is sort finite, as we do not have any relabelling as a construction in the language. The operational semantics for a $p \in Trees$ is defined as the process graph (p, Λ_{Tree}). For $d \in FinTrees$ the process graph that gives its semantics may be represented as an element of ST. Thus the operational semantics for d is given by $\mathcal{G}(d)$ obtained by the following recursive definition:

1. $\mathcal{G}(nil) = \emptyset$,
2. $\mathcal{G}(\Omega) = \{\perp\}$,
3. $\mathcal{G}(\mu.d) = \mu : \mathcal{G}(d)$,
4. $\mathcal{G}(d_1 + d_2) = \mathcal{G}(d_1) \oplus \mathcal{G}(d_2)$.

Of course the definitions of $\underset{\sim}{\sqsubset}$, $\underset{\approx}{\sqsubset}$, $\underset{\approx}{\sqsubset}^c$ and $\underset{\approx}{\sqsubset}^*$ and their ω-versions apply for the lts Λ_{Tree} and as before we have that $\underset{\approx}{\sqsubset}^c = \underset{\approx}{\sqsubset}^*$ and $\underset{\approx_\omega}{\sqsubset}^c = \underset{\approx_\omega}{\sqsubset}^*$.

In [Wal90] and [AH92] the preorder $\underset{\approx_\omega}{\sqsubset}^*$ is given an equational characterizations in terms of equationally based proof systems. In Figures 1 and 2 we define such a proof system for $Trees$, which is a slight modification of the proof systems in the afore mentioned references. The proof system consists of a set of inequations, Figure 1, and a set of inference rules, Figure 2. We refer to the full proof system as E_{rec} but the sub-system where the rules (ω) and (rec) are omitted we call E. We write $\sqsubseteq_{E_{rec}}$ and $\sqsubseteq_E$ for the induced preorders. The syntactic approximations p^n, that occur in the rule (ω), are also standard (see e.g. [Hen88b]) and are defined inductively as follows:

Definition 19. (Finite Syntactical Approximations)

1. $u^0 = \Omega$ for all $u \in TreeTerms$.
2. (a) $nil^{n+1} = nil$, $\Omega^{n+1} = \Omega$ and $x^{n+1} = x$ for $x \in \texttt{Var}$,
   (b) $(u_1 + u_2)^{n+1} = u_1^{n+1} + u_2^{n+1}$,
   (c) $(\mu.u)^{n+1} = \mu.(u^{n+1})$,
   (d) $(\texttt{rec}x.u)^{n+1} = u^{n+1}[(\texttt{rec}x.u)^n/x]$.

Here we note that if $p \in Trees$ then $p^n \in FinTrees$. We get the following soundness and completeness result as a special case of the more general soundness and completeness theorem in [AH92].

A1 $x + y = y + x$	$\boldsymbol{\Omega 1}$ $\Omega \sqsubseteq x$
A2 $x + (y + z) = (x + y) + z$	$\boldsymbol{\Omega 2}$ $\tau.(x + \Omega) \sqsubseteq x + \Omega$
A3 $x + x = x$	$\boldsymbol{\tau 1}$ $\mu.\tau.x = \mu.x$
A4 $x + nil = x$	$\boldsymbol{\tau 2}$ $\tau.x + x = \tau.x$
	$\boldsymbol{\tau 3}$ $\mu.(x + \tau.y) = \mu.(x + \tau.y) + \mu.y$

Fig. 1. The Inequations

(ref)	$p \sqsubseteq p$	$(trans)$	$\dfrac{p \sqsubseteq q,\ q \sqsubseteq r}{p \sqsubseteq r}$
(pre)	$\dfrac{p \sqsubseteq q}{\mu.p \sqsubseteq \mu.q}$	(sum)	$\dfrac{p_1 \sqsubseteq p_2, q_1 \sqsubseteq q_2}{p_1 + q_1 \sqsubseteq p_2 + q_2}$
(rec)	$\dfrac{}{\mathtt{rec}P.p = p[\mathtt{rec}P.p/P]}$	(ω)	$\dfrac{p^{(n)} \sqsubseteq q \text{ for all } n}{p \sqsubseteq q}$

$(inst)$ $\dfrac{}{p \sqsubseteq q}$ $p \sqsubseteq q$ is a closed instantiation of the inequations in E

Fig. 2. The Proof system E_{rec}

Theorem 20. *The proof system E_{rec} is sound and complete for $Trees$ with respect to the preorder $\sqsubseteq^c_{\approx_\omega}$.*

From [Wal90] we borrow the following notion of *syntactic normal forms.*

Definition 21. (Syntactic Normal Forms) We say that $\eta \in FinTrees$ is a normal form if $\eta = \sum_i \mu_i.\eta_i[+\Omega]$ and

1. each η_i is a syntactic normal form,
2. if $\mu_i = \tau$ then $\eta_i \Downarrow$
3. if $\eta \Downarrow$ and $\eta \Uparrow a$ then $a.\Omega$ is a summand of η.
4. if $\eta \stackrel{\mu}{\Longrightarrow} \eta'$ then $\eta \stackrel{\mu}{\longrightarrow} \eta'$.

We denote the set of syntactic normal forms by NF ranged over by η.

The following lemma gives the relationship between the syntactic and the semantic normal forms.

Lemma 22. $\eta \in NF$ *iff* $\mathcal{G}(\eta) \in \mathsf{NST}$.

Proof. Follows from a simple structural induction on η.

The following result is proved in [Wal90].

Theorem 23 Normalization Theorem. *For all* $d \in FinTree$ *there is* $\eta \in NF$ *such that* $d =_E \eta$.

4 Preliminaries on Algebraic Semantics

In this section, we review the basic notions of algebraic semantics and domain theory that will be needed in the remainder of this study. We assume that the reader is familiar with the basic notions of ordered and continuous algebras (see, e.g., [Gue81, Hen88a, AJ95]); however, in what follows we give a quick overview of the way a denotational semantics can be given to a recursive language like $Trees$ following the standard lines of algebraic semantics [Gue81]. The interested reader is invited to consult [Hen88a] for an explanation of the theory.

In what follows, we let Σ denote a signature, i.e. a set of syntactic operators provided with a function, *arity*: $\Sigma \longrightarrow Nat$ which gives the number of arguments the operator takes. A Σ-algebra is a pair $(\mathcal{A}, \Sigma_\mathcal{A})$, where $\mathcal{A}$ is the *carrier* set and $\Sigma_\mathcal{A}$ is a set of *semantic operators* $f_\mathcal{A} : \mathcal{A}^l \to \mathcal{A}$, where $f \in \Sigma$ and $l = arity(f)$. We call $f_\mathcal{A}$ the *interpretation* of the syntactic operator f in $\mathcal{A}$. Let $(\mathcal{A}, \Sigma_\mathcal{A})$ and $(\mathcal{B}, \Sigma_\mathcal{B})$ be Σ-algebras. A mapping $\varphi : \mathcal{A} \to \mathcal{B}$ is a *Σ-homomorphism* if it preserves the Σ-structure, i.e. if for every $f \in \Sigma$ and vector $\overline{a}$ of elements of $\mathcal{A}$ of the length $arity(f)$:

$$\varphi(f_\mathcal{A}(\overline{a})) = f_\mathcal{B}(\varphi(\overline{a})) \ .$$

The *term algebra* $T(\Sigma)$ is the *initial* Σ-algebra, i.e., if $(\mathcal{A}, \Sigma_\mathcal{A})$ is a Σ-algebra then there is a unique Σ-homomorphism $\iota_\mathcal{A} : T(\Sigma) \to \mathcal{A}$. We refer to this homomorphism as the *interpretation* of $T(\Sigma)$ in $\mathcal{A}$. We write $T(\Sigma, \mathtt{Var})$ for the term algebra that contains the set of variables $\mathtt{Var}$ as operators of arity 0 and $T^{rec}(\Sigma, \mathtt{Var})$ if it also allows the recursion construction $\mathtt{rec}x.t$.

The obvious idea is to model a language like that of $Trees$ by a Σ-algebra where Σ is the set of finite term-forming operations (in the example mentioned, we get $\Sigma = \{\Omega, nil, +\} \cup \{\mu.\_ \mid \mu \in Act_\tau\}$). However, this is not sufficient to model operations like recursion. For that, we need to consider a slightly more sophisticated concept.

A *Σ-domain* $(\mathcal{A}, \sqsubseteq_\mathcal{A}, \Sigma_\mathcal{A})$ is a Σ-algebra whose carrier $(\mathcal{A}, \sqsubseteq_\mathcal{A})$ is an algebraic complete partial order (cpo) (see e.g. [Plo81]) and whose operators in $\Sigma_\mathcal{A}$ are continuous. The notion of *Σ-poset* (respectively *Σ-preorder*) may be defined in a similar way by requiring that $(\mathcal{A}, \sqsubseteq_\mathcal{A})$ is a partially ordered (resp. preordered) set and that the operators are monotonic. The notion of Σ-homomorphism extends to the ordered Σ-structures in the obvious way by requiring that such maps preserve the underlying order-theoretic structure as well as the Σ-structure. Any

Σ-preorder induces a unique Σ-poset which we refer to as its kernel. For any Σ-algebra, $\mathcal{A}$, ordered or not, the set $(\mathtt{Var} \to \mathcal{A})$ of $\mathcal{A}$-environments will be denoted by $\mathrm{ENV}_{\mathcal{A}}$, and ranged over by the meta-variable ρ. The (unique) interpretation of $T(\Sigma, \mathtt{Var})$ in $\mathcal{A}$ is the mapping $\mathcal{A}[\![\cdot]\!] : T(\Sigma, \mathtt{Var}) \to (\mathrm{ENV}_{\mathcal{A}} \to \mathcal{A})$ defined recursively by:

$$\begin{aligned} \mathcal{A}[\![x]\!]\rho &\triangleq \rho(x), \\ \mathcal{A}[\![f(p_1, \ldots, p_l)]\!]\rho &\triangleq f_{\mathcal{A}}(\mathcal{A}[\![p_1]\!]\rho, \ldots, \mathcal{A}[\![p_l]\!]\rho). \end{aligned}$$

If $\mathcal{A}$ is a Σ-domain the interpretation extends to the term algebra $T^{rec}(\Sigma, \mathtt{Var})$ by setting

$$\mathcal{A}[\![\mathtt{rec} x.u]\!]\rho \triangleq \mathsf{Y}\lambda a.\, \mathcal{A}[\![u]\!]\rho[x \to a]$$

where Y denotes the least fixed-point operator. As usual, $\rho[x \to a]$ denotes the environment which is defined as follows:

$$\rho[x \to a](y) \triangleq \begin{cases} a & \text{if } x = y \\ \rho(y) & \text{otherwise.} \end{cases}$$

Note that, for each closed recursive term $p \in T^{rec}(\Sigma, \mathtt{Var})$, $\mathcal{A}[\![p]\!]\rho$ does not depend on the environment ρ. The denotation of a closed term, p, will be denoted by $\mathcal{A}[\![p]\!]$. For recursion free closed terms the mapping $\mathcal{A}[\![\_]\!]$ coincides with $\iota_{\mathcal{A}}$.

It is worth pointing out that the initial Σ-algebra for a set of generators, $\mathtt{Var}$, does indeed exist. Actually, more than just that is true: We can also require a set of equations to hold on the resulting Σ-algebra (such as $x + x = x$, for example). The initial Σ-algebra for a set of generators satisfying a given set of equations is constructed from the term-model by defining an equivalence relation on terms. The operations are well defined with respect to the equivalence classes so that the resulting quotient is again a Σ-algebra.

To find the initial ordered Σ-algebras, we have to say how to construct them. For posets, the process is very much like that of constructing the initial Σ-algebra - only this time one can actually start with a poset of generators, and the order for the resulting Σ-poset is then defined recursively on the terms such that the operations become monotonic. We can even do more in that case: instead of giving a set of equations which we want to hold, we can now deal with a set of inequalities. A typical inequality that one wants to hold in models for languages like ours is $\Omega \leq x$ which can thus be built in. We are, however, not interested in the initial Σ-poset for a set of generators and inequalities but in the initial Σ-domain. These two, however, are closely related: The initial Σ-domain can be obtained as the ideal completion of the corresponding Σ-poset - the operations are the unique continuous extensions of the corresponding operators for the Σ-poset. Similarly any Σ-preorder induces a unique Σ-domain; the ideal completion of its kernel. For more details on how this works, see Chapter 6 in [AJ95].

5 A Fully Abstract Denotational Model for *Trees*

In this section we will define a Σ-domain (where Σ consists of the operators nil, Ω, $+$ and $\mu.\_, (\mu \in Act_\tau)$) in such a way that the derived denotational semantics for $Trees$ is fully abstract with respect to observational precongruence. In particular this means that the operator Ω has to be interpreted as the bottom element of the domain as it is a least element with respect to the observational precongruence.

We show the the full abstractness of the model by showing that it is the initial Σ-algebra with respect to the operations in our language plus the inequations in E. We obtain this by proving that the proof system E, interpreted in the model, is sound and complete with respect to the preorder of the model. The full abstractness then follows from the fact that the proof system is sound and complete with respect to the observational precongruence as stated in Theorem 20.

The domain is obtained as an abstraction on the preorder $(\mathit{NST}, \sqsubseteq_{\approx}^{*})$ as explained in the Introduction.

Definition 24. We define the Σ-preorder $(\mathit{NST}, \sqsubseteq_{\mathit{NST}}, \Sigma_{\mathit{NST}})$ as follows:

1. The preorder $\sqsubseteq_{\mathit{NST}}$ is defined as the least binary relation over NST satisfying:

$$\begin{array}{ll} n \sqsubseteq_{\mathit{NST}} m \text{ if} & (1)\ \langle \mu, n' \rangle \in n \Rightarrow \exists \langle \mu, m' \rangle \in m.\ n' \sqsubseteq_{\mathit{NST}} m' \ \text{ and} \\ & (2)\ \bot \in m \Rightarrow \bot \in n \ \text{ and} \\ & (3)\ \langle \mu, m' \rangle \in m \Rightarrow (\bot \in n \text{ or } \exists \langle \mu, n' \rangle \in n.\ n' \sqsubseteq_{\mathit{NST}} m') \end{array}$$

2. The structure Σ_{NST} is defined as follows:
   (a) $\Omega_{\mathit{NST}} = \{\bot\}$,
   (b) $nil_{\mathit{NST}} = \emptyset$,
   (c) μ_{NST}: (compare Definition 10)

$$\tau_{\mathit{NST}}.n = \begin{cases} n & \text{if } \bot \in n \\ \{\langle \tau, n \rangle\} \cup n & \text{if } \bot \notin n \end{cases}$$

$$a_{\mathit{NST}}.n = \begin{cases} \{\langle a, n \rangle\} \cup \{\langle a, n' \rangle | \langle \tau, n' \rangle \in n\} & \text{if } \bot \notin n \\ \{\langle a, n \rangle\} \cup \{\langle a, n' \rangle | \langle \tau, n' \rangle \in n\} \cup \{\langle a, \bot \rangle\} & \text{if } \bot \in n \end{cases}$$

   (d) $+_{\mathit{NST}}$: $n_1 +_{\mathit{NST}} n_2 = n_1 \cup n_2$

Now we have:

Lemma 25.

1. *The preorders $\sqsubseteq_{\mathsf{NST}}$ and $\sqsubseteq_{\approx}$ coincide on* NST.
2. $(\mathsf{NST}, \sqsubseteq_{\mathsf{NST}}, \Sigma_{\mathsf{NST}})$ *is a Σ-preorder.*
3. *For all $\eta \in NF$,* $\mathsf{NST}[\![\eta]\!] = \mathcal{G}(\eta)$.

Proof.

1. Is proved in [Abr91].

2. Follows directly from Lemma 12.1.
3. Follows by Lemma 12.2 and a simple structural induction on η.

Part 3. of the lemma above says that the denotational interpretation in NST of a normal form is exactly its operational semantics, i.e. the process graph that is generated by the rules for the operational semantics for the language $Trees$.

Unfortunately the equation ($\boldsymbol{\tau 1}$) is not sound in NST as the following example shows. (To ease the notational complexity we use the notation $\mu : n$ to denote $\{\langle \mu, n\rangle\}$ in what follows as explained in Section 2.)

Example 2. Let $n = \emptyset$. Then

$$\tau_{\mathsf{NST}}.\emptyset = \tau : \emptyset \cup \emptyset = \tau : \emptyset$$

and therefore

$$\tau_{\mathsf{NST}}.\tau_{\mathsf{NST}}.\emptyset = \tau : \tau : \emptyset \cup \tau : \emptyset.$$

It is easy to see that

$$\tau_{\mathsf{NST}}.\tau_{\mathsf{NST}}.\emptyset \not\sim \tau_{\mathsf{NST}}.\emptyset.$$

However we have the following partial soundness result and a completeness result. Let F denote the proof system E minus the equation ($\boldsymbol{\tau 1}$). Then we have:

Lemma 26. *The proof system F is sound and complete for* $(\mathsf{NST}, \sqsubseteq_{\mathsf{NST}}, \Sigma_{\mathsf{NST}})$.

Proof. The soundness of the inequations **(*A*1)**–**(*A*4)** and ($\boldsymbol{\Omega 1}$)–($\boldsymbol{\Omega 2}$) is obvious. The soundness of the inference rules follows from the fact that $(\mathsf{NST}, \sqsubseteq_{\mathsf{NST}}, \Sigma_{\mathsf{NST}})$ is a Σ-preorder. What remains to prove is the soundness of ($\boldsymbol{\tau 2}$) and ($\boldsymbol{\tau 3}$). We proceed as follows:

($\boldsymbol{\tau 2}$): Assume that $n \in \mathsf{NST}$. We will prove that

$$\tau_{\mathsf{NST}}.n +_{\mathsf{NST}} n =_{\mathsf{NST}} n.$$

The case when $\perp \in n$ is obvious so we may assume that $\perp \notin n$. Then we have

$$\tau_{\mathsf{NST}}.n +_{\mathsf{NST}} n = (\tau : n \cup n) \cup n = \tau : n \cup n = \tau_{\mathsf{NST}}.n.$$

($\boldsymbol{\tau 3}$) Assume that $n_1, n_2 \in \mathsf{NST}$, we will show that

$$\mu_{\mathsf{NST}}.(n_1 +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2) =_{\mathsf{NST}} \mu_{\mathsf{NST}}.(n_1 +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2) + \mu_{\mathsf{NST}}.n_2$$

We have the two possible cases: $\mu = \tau$ and $\mu \neq \tau$. We proceed as follows:

$\mu = \tau$: The case when $\perp \in n_1 \cup n_2$ is straight forward and is left to the reader. Next let us assume that $\perp \notin n_1 \cup n_2$. Then

$$\begin{aligned} &\tau_{\mathsf{NST}}.(n_1 +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2) \\ &= \tau : (n_1 \cup \tau : n_2 \cup n_2) \cup (n_1 \cup \tau : n_2 \cup n_2) \\ &= \tau : (n_1 \cup \tau : n_2 \cup n_2) \cup (n_1 \cup \tau : n_2 \cup n_2) \cup (\tau : n_2 \cup n_2) \\ &= \tau_{\mathsf{NST}}.(n_1 +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2) +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2. \end{aligned}$$

$\mu = a \in Act$: Again we have two possible sub-cases: $\bot \in n_2$ and $\bot \notin n_2$.

$\bot \in n_2$: First we note that

$$\bot \in n_2 \text{ implies } n_2 \sqsubseteq_{\mathsf{NST}} n_1 \cup n_2. \tag{3}$$

Now we have the following:

$$\begin{aligned}
& a_{\mathsf{NST}}.(n_1 +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2) \\
= \quad & a : (n_1 \cup n_2) \cup \bigcup\{a : n' | \tau : n' \in n_1 \cup n_2\} \cup a : \{\bot\} \\
=_{\mathsf{NST}} \; & a : (n_1 \cup n_2) \cup \bigcup\{a : n' | \tau : n' \in n_1 \cup n_2\} \cup a : \{\bot\} \cup \\
& a : n_2 \cup \bigcup\{a : n_2' | \tau : n_2' \in n_2\} \cup a : \{\bot\} \\
& \quad (\text{ as } a : \{\bot\} \sqsubseteq_{\mathsf{NST}} a : n_2 \sqsubseteq_{\mathsf{NST}} a : (n_1 \cup n_2) \text{ by } (3)) \\
= \quad & a_{\mathsf{NST}}.(n_1 +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2) +_{\mathsf{NST}} a_{\mathsf{NST}}.n_2
\end{aligned}$$

$\bot \notin n_2$: We proceed as follows:

$$\begin{aligned}
& a_{\mathsf{NST}}.(n_1 +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2) \\
&= a_{\mathsf{NST}}.(n_1 \cup \tau : n_2 \cup n_2) \\
&= a : (n_1 \cup \tau : n_2 \cup n_2) \cup \bigcup\{a : n' | \tau : n' \in n_1 \cup n_2\} \cup \\
&\quad\; a : n_2 \cup \{a : \{\bot\} | \bot \in n_1\} \\
&= a : (n_1 \cup \tau : n_2 \cup n_2) \cup \bigcup\{a : n' | \tau : n' \in n_1 \cup n_2\} \cup \\
&\quad\; a : n_2 \cup \{a : \{\bot\} | \bot \in n_1\} \cup \\
&\quad\; a : n_2 \cup \bigcup\{a : n' | \tau : n' \in n_2\} \\
&= a_{\mathsf{NST}}.(n_1 +_{\mathsf{NST}} \tau_{\mathsf{NST}}.n_2) +_{\mathsf{NST}} a_{\mathsf{NST}}.n_2.
\end{aligned}$$

Here we note that $=_{\mathsf{NST}}$ appears only once in the sequence of the proof above. All the other equalities are set equalities.

The completeness may be easily proved by induction on the combined depth of n and m using Lemma 12. (In fact we do not need the $\boldsymbol{\tau 2}$,$\boldsymbol{\tau 3}$ and $\boldsymbol{\Omega 2}$ at all to prove the completeness as the preorder $\sqsubseteq_{\mathsf{NST}}$ coincides with the strong bisimulation preorder $\sqsubseteq\!\!\!\!\sim$.)

To obtain a Σ-preorder where the equation $\boldsymbol{\tau 1}$ is sound, we follow the suggestion after Example 1. This leads to the following definition (where $=_{\tau 1}$ has the same meaning as in Section 2).

Definition 27. We define $(K, \sqsubseteq_K, \Sigma_K)$ as follows:

1. $K = \{[n] \mid [n] = \{n' \in \mathsf{NST} | n' =_{\tau 1} n\}\}$.
2. $\sqsubseteq_K$ is defined by

$$[n_1] \sqsubseteq_K [n_2] \text{ iff } \exists n_1', n_2' \in \mathsf{NST}. n_1' =_{\tau 1} n_1, n_2' =_{\tau 1} n_2 \text{ and } n_1' \sqsubseteq_{\mathsf{NST}} n_2'.$$

3. Σ_K is defined by
   (a) $nil_K = [\emptyset] = \{\emptyset\}$,
   (b) $\Omega_K = [\{\bot\}] = \{\{\bot\}\}$,
   (c) $[n_1] +_K [n_2] = [n_1 +_{NST} n_2]$,
   (d) $\mu_K.[n] = [\mu_{NST}.n]$.

Here we would like to point out that this model is not a term model as the elements are not equivalence classes over syntactic terms but process graphs. Furthermore the model K is not obtained by simply factoring out equations but involves operational reasoning as well. Now we have the following lemma:

Lemma 28.

1. *For all $n_1, n_2 \in \mathsf{NST}$, $[n_1] \sqsubseteq_K [n_2]$ iff $n_1 \underset{\approx}{\sqsubseteq}^* n_2$.*
2. *$(K, \sqsubseteq_K, \Sigma_K)$ is a $\Sigma-preorder$.*
3. *The proof system E is sound and complete on $(K, \sqsubseteq_K, \Sigma_K)$.*

Proof. It is easy to see that $\sqsubseteq_K$ is well defined this way, i.e. is independent of the representants for the classes $[n_1]$ and $[n_2]$. As $\sqsubseteq_{NST} = \underset{\sim}{\sqsubseteq}$ on NST the first statement follows from the Characterization Theorem 16, and the fact that $=_{\tau 1} \subseteq \approx^*$. This in turn ensures that $\sqsubseteq_K$ is a preorder. To prove statement 2. it only remains to prove that the operators in Σ_K are well defined and monotonic. This is an easy consequence of the way they are defined and the fact that $=_{\tau 1}$ is preserved by the operators in NST. What remains to proof is therefore statement 3., the soundness and the completeness of the proof system E on $(K, \sqsubseteq_K, \Sigma_K)$. To prove this we proceed as follows:

Soundness: The only non trivial case is the soundness of $\boldsymbol{\tau 1}$. So assume $n \in NST$ and we will prove that $\mu_K.\tau_K.[n] =_K \mu_K.[n]$. We recall that by definition of $=_{\tau 1}$,

$$\mu_{NST}.\tau_{NST}.n =_{\tau 1} \mu_{NST}.n.$$

This implies

$$\mu_K.\tau_K.[n] =_K [\mu_{NST}.(\tau_{NST}.n)] =_K$$

$$[\mu_{NST}.n] =_K \mu_K.[n].$$

Completeness: First we note that $=_{\tau 1} \subseteq =_E$. Next let $n_1, n_2 \in NST$ and we have the following:

$$[n_1] \sqsubseteq_K [n_2]$$

$$\text{implies } \exists n_1', n_2' \in NST.\ n_1 =_{\tau 1} n_1' \sqsubseteq_{NST} n_2' =_{\tau 1} n_2 \quad \text{(by definition)}$$

$$\text{implies } \exists n_1', n_2' \in NST.\ n_1 =_E n_1' \sqsubseteq_E n_2' =_E n_2 \quad \text{(by Lem. 26)}$$

$$\text{implies } n_1 \sqsubseteq_E n_2.$$

We have the following result:

Lemma 29.

1. *For all* $d \in FinTree$, $K[\![d]\!] = [\mathsf{NST}[\![d]\!]]$.
2. *For each* $k \in K$ *there is an* $\eta \in NF$ *such that* $K[\![\eta]\!] = k$.

Proof.

1. The mapping $[\mathit{NST}[\![\_]\!]]$ is an interpretation of $FinTrees$ in K. The result follows by uniqueness of such mappings.
2. By definition of K, $k = [n]$ for some $n \in \mathit{NST}$. By a simple induction on the depth of n we may show that there exists an $\eta \in NF$ such that $\mathit{NST}[\![\eta]\!] = n$. By part 1. we get
$$K[\![\eta]\!] = [\mathit{NST}[\![\eta]\!]] = k.$$

5.1 Soundness, Completeness and Full Abstractness for *Trees*

We complete the construction of the the full domain by taking $(\overline{K}, \sqsubseteq_{\overline{K}}, \Sigma_{\overline{K}})$ to be the unique Σ-domain generated by $(K, \sqsubseteq_K, \Sigma_K)$ as described in Chapter 4. The following theorem is standard and is proved in e.g. [Hen88b].

Theorem 30. *For all* $p \in Trees$, $\overline{K}[\![p]\!] = \bigsqcup_n \overline{K}[\![p^n]\!]$.

Now we have the following equivalence result:

Theorem 31. *For all* $p_1, p_2 \in Trees$
$$\overline{K}[\![p_1]\!] \sqsubseteq_{\overline{K}} \overline{K}[\![p_2]\!] \textit{ iff } p_1 \sqsubseteq_{E_{rec}} p_2 \textit{ iff } p_1 \mathrel{\underset{\approx}{\sqsubset}}^* p_2.$$

Proof. That $p_1 \sqsubseteq_{E_{rec}} p_2$ iff $p_1 \mathrel{\underset{\approx}{\sqsubset}}^* p_2$ is the content of Theorem 20. Therefore we only have to prove that $\overline{K}[\![p_1]\!] \sqsubseteq_{\overline{K}} \overline{K}[\![p_2]\!]$ iff $p_1 \sqsubseteq_{E_{rec}} p_2$, i.e. that the proof system E_{rec} is sound and complete with respect to the denotational model.

Soundness: The soundness of the (ω)-rule is the content of Theorem 30 whereas the soundness of the (rec)-rule follows from the definition of the semantics of $\mathtt{rec}.p$ as the least fixed point. It remains to prove the soundness of E. To prove this we first prove by a simple induction on the depth of the proof for $p \sqsubseteq_E q$ that
$$p \sqsubseteq_E q \Rightarrow \forall n. p^n \sqsubseteq_E q^n. \tag{4}$$
Then we note that the soundness of E over $FinTrees$ with respect to K follows directly from the soundness of E in K. To prove the general result, i.e. the soundness of E over $Trees$ with respect to $\overline{K}$, we may proceed as follows. Assume $p \sqsubseteq_E q$. Then, by (4), $p^n \sqsubseteq_E q^n$ for all n. As $p^n, q^n \in FinTrees$, the soundness of E with respect to K implies
$$K[\![p^n]\!] \sqsubseteq_K K[\![q^n]\!] \text{ for all } n,$$
or equivalently
$$\overline{K}[\![p^n]\!] \sqsubseteq_{\overline{K}} \overline{K}[\![q^n]\!] \text{ for all } n.$$
Finally Theorem 30 implies
$$\overline{K}[\![p]\!] \sqsubseteq \overline{K}[\![q]\!].$$

Completeness: Again we reduce the proof to proving that E is complete over $FinTrees$ with respect to K. We first note that Theorem 30 and the ω-algebraicity of the model imply

$$\begin{array}{ll} \overline{K}[\![p]\!] \sqsubseteq_{\overline{K}} \overline{K}[\![q]\!] & \text{implies} \\ \forall n. \overline{K}[\![p^n]\!] \sqsubseteq_{\overline{K}} \overline{K}[\![q]\!] & \text{implies} \\ \forall n \exists m. \overline{K}[\![p^n]\!] \sqsubseteq_{\overline{K}} \overline{K}[\![q^m]\!] & \text{implies} \\ \forall n \exists m. K[\![p^n]\!] \sqsubseteq_K K[\![q^m]\!]. & \end{array} \tag{5}$$

If E is complete over $FinTrees$ with respect to K then

$$K[\![p^n]\!] \sqsubseteq_K K[\![q^m]\!] \text{ implies } p^n \sqsubseteq_E q^m. \tag{6}$$

Now $q^m \sqsubseteq_{E_{rec}} q$ may easily be shown so (5), (6) and the ω-rule give

$$\overline{K}[\![p]\!] \sqsubseteq_{\overline{K}} \overline{K}[\![q]\!] \text{ implies } \forall n. p^n \sqsubseteq_{E_{rec}} q \text{ implies } p \sqsubseteq_{E_{rec}} q.$$

So it only remains to prove the completeness of E over $FinTrees$ with respect to K. Furthermore, by the the normalization Theorem 23, it is sufficient to prove the completeness over normal forms. To prove this completeness result we proceed as follows:
Assume $\eta_1, \eta_2 \in NF$. By Lemma 25 and Lemma 29

$$K[\![\eta_i]\!] = [\mathsf{NST}[\![\eta_i]\!]] = [\mathcal{G}(\eta_i)]$$

for $i = 1, 2$. Therefore we have:

$$\begin{array}{lll} K[\![\eta_1]\!] \sqsubseteq_K K[\![\eta_2]\!] & & \\ \text{iff} \;\; [\mathcal{G}(\eta_1)] \sqsubseteq_K [\mathcal{G}(\eta_2)] & & \\ \text{iff} \;\; \mathcal{G}(\eta_1) \sqsubseteq^*_{\approx} \mathcal{G}(\eta_2) & & \text{by Lem. 28} \\ \text{iff} \;\; \eta_1 \sqsubseteq^*_{\approx} \eta_2 & \text{by definition of the op. sem.} & \\ \text{iff} \;\; \eta_1 \sqsubseteq_E \eta_2 & \text{as } E \text{ is complete wrt. } \sqsubseteq^*_{\approx}. & \end{array}$$

6 Conclusion and Future Work

Regarding the picture being drawn in the introduction about ways of getting fully abstract denotational models for concurrent languages with an observational preorder, we have obtained the following: By giving a set of inequations, we have found a way of having a term model. Also, we have constructed a syntax free model which is the ideal completion of a preordered set whose elements are finite synchronization trees like the ones that appear in the representation of Abramsky's model for strong bisimulation preorder. These trees are a representation of transition graphs in normal forms which in turn are derived as the operational semantics for syntactic normal forms in the sense of [Wal90]. By defining the operators in a suitable way we obtained a Σ-preorder. Unfortunately the Σ-domain obtained directly as an ideal closure of this Σ-preorder does not satisfy the set of equations that characterize the ω-observational congruence as the equation $\mu.\tau.x = \mu.x$ is not sound in this domain. We obtain a fully abstract model as a further abstraction of this model; roughly speaking we factor out the missing equation and obtain a fully abstract model with respect to the behavioural preorder we had in mind.

What is still missing, is the last part: Finding a mathematical description of the model which does not mention equations at all. This has proved to be more difficult than we first expected. To illustrate the kind of difficulties one runs into, let us consider a related successful attempt of doing something like this. In [Abr91], Abramsky defines a fully abstract denotational model for synchronization trees with respect to strong bisimulation precongruence. It is given as the solution of the recursive domain equation

$$D \cong \mathcal{P}(\sum_{\mu \in Act_\tau} D)$$

where $\mathcal{P}$ is a variant of the Plotkin (or convex) power construction (including the empty set) and $\sum$ is a lifted disjoint union. His proof that this is indeed a fully abstract model for the feature in question uses a sophisticated mathematical machinery that also produces a logic to reason about the model. There is however a shortcut to convince oneself that this is what one wants: Strong bisimulation precongruence for the language we consider can be characterized in terms of the equations **A1** to **A4** which say that $+$ is idempotent, symmetric, associative and has a unit. There are no (in)equations concerning prefixing and in particular no (in)equations connecting prefixing with $+$. It is then not hard to see that the initial solution to the above domain equation is exactly the free Σ-domain for the empty set of generators where the equations **A1** to **A4** hold and where the set of operators Σ contains $+$ as well as a unary operator for every element of Act_τ. The fact that this Σ-domain can be presented in such an appealing way crucially depends on the simplicity of the equations describing the modelled precongruence.

Since the domain we are looking for is the free Σ-domain for the same set of actions but with the additional inequations $\boldsymbol{\Omega}\mathbf{1}$, $\boldsymbol{\Omega}\mathbf{2}$ and $\boldsymbol{\tau}\mathbf{1}$ to $\boldsymbol{\tau}\mathbf{3}$, the domain we are looking for is a quotient of the one given by Abramsky (the mathematical details of this process can be found in [AJ95]). The domain we give as a model is an improvement over just taking the quotient of Abramsky's, since the only thing we have to take extra trouble over is inequality $\boldsymbol{\tau}\mathbf{1}$. However, it is not quite what we had in mind as forming quotients of this kind is a somewhat obscure process. It is our aim to find another way of presenting this Σ-domain, if possible also as the solution of a recursive domain equation.

A Appendix

In what follows we will prove Theorem 14. For this purpose we need the following definition.

Definition 32 [Wal90]. Given $\mathcal{R} \in \mathsf{Rel(P)}$ we define $s_1 \mathcal{R}^\diamond s_2$ by:
$s_1 \mathcal{R}^\diamond s_2$ iff

1. if $s_1 \xrightarrow{\mu} s_1'$ then, for some s_2', $s_2 \xrightarrow{\mu} s_2'$ and $s_1' \mathcal{R} s_2'$
2. if $s_1 \downarrow$ then
   (a) $s_2 \downarrow$
   (b) if $s_2 \xrightarrow{\mu} s_2'$ then, for some s_1', $s_1 \xrightarrow{\mu} s_1'$ and $s_1' \mathcal{R} s_2'$

Theorem 14 is a direct consequence of the following Lemma.

Lemma 33. *For all $n_1, n_2 \in \mathsf{NST}$*

1. $n_1 \underset{\approx}{\sqsubset} n_2 \quad \Leftrightarrow \quad n_1 \underset{\approx}{\sqsubset}_g n_2$,
2. $n_1 \underset{\approx}{\sqsubset}^* n_2 \quad \Leftrightarrow \quad n_1 \underset{\approx}{\sqsubset}_g^\diamond n_2$,

Proof.

1. The "$\Longleftarrow$" part is proved in [Wal90, Lemma 7]. Therefore we only have to concentrate on proving the "$\Longrightarrow$" part, i.e that $\underset{\approx}{\sqsubset} \subseteq \underset{\approx}{\sqsubset}_g$. By definition
$$\underset{\approx}{\sqsubset}_g = \bigcup \{\mathcal{R} | \mathcal{R} \subseteq \mathcal{F}_w^g(\mathcal{R})\}.$$
It is therefore sufficient to prove that $\underset{\approx}{\sqsubset} \subseteq \mathcal{F}_w^g(\underset{\approx}{\sqsubset})$. First we recall that for a normal form n, $n \overset{a}{\Longrightarrow} n'$ iff $n \xrightarrow{a} n'$ and $n \Downarrow$ iff $n \downarrow$. Now we proceed as follows: Assume $m_1 \underset{\approx}{\sqsubset} m_2$. As $\underset{\approx}{\sqsubset}$ is a fixed point to $\mathcal{F}_w$ then $m_1 \mathcal{F}_w(\underset{\approx}{\sqsubset}) m_2$. We will prove that $m_1 \mathcal{F}_w^g(\underset{\approx}{\sqsubset}) m_2$.
   (a) If $m_1 \xrightarrow{a} m_1'$, then there is a m_2' such that $m_2 \xrightarrow{a} m_2'$ and $m_1' \underset{\approx}{\sqsubset} m_2'$. Similarly if $m_1 \xrightarrow{\tau} m_1'$, then there is an m_2' such that $m_2 \overset{\varepsilon}{\Longrightarrow} m_2'$ and $m_1' \underset{\approx}{\sqsubset} m_2'$.
   (b) Assume $m_1 \downarrow$, then $m_2 \downarrow$ as $m_1 \underset{\approx}{\sqsubset} m_2$.
   (c) Assume that $m_1 \downarrow$, $m_2 \downarrow$ and $m_2 \xrightarrow{a} m_2'$. We have the following cases:
      $m_1 \Downarrow a$: As $m_1 \underset{\approx}{\sqsubset} m_2$ then $m_2 \Downarrow a$ and $m_1 \xrightarrow{a} m_1'$ for some m_1' such that $m_1' \underset{\approx}{\sqsubset} m_2'$.
      $m_1 \Uparrow a$: As $m_1 \downarrow$ this implies that $\langle a, \{\bot\}\rangle \in m_1$. Therefore $m_1 \xrightarrow{a} \{\bot\}$ where $\{\bot\} \underset{\approx}{\sqsubset} m_2'$.
   (d) Finally assume $m_1 \downarrow$, $m_2 \downarrow$ and $m_2 \xrightarrow{\tau} m_2'$. Then there exists a m_1' such that $m_1 \overset{\varepsilon}{\Longrightarrow} m_1'$ and $m_1' \underset{\approx}{\sqsubset} m_2'$.
2. By part 1 it is sufficient to prove that
$$n_1 \underset{\approx}{\sqsubset}_g^* n_2 \quad \Leftrightarrow \quad n_1 \underset{\approx}{\sqsubset}_g^\diamond n_2$$
We only prove the "$\Longleftarrow$" part as the "$\Longrightarrow$" part may be proved in the same way as the "$\Longrightarrow$" part for the previous case, part 1. Assume $n_1 \underset{\approx}{\sqsubset}_g^\diamond n_2$.

(a) Assume $n_1 \xrightarrow{a} n_1'$. As $n_1 \sqsubseteq^{\diamond}_{\approx g} n_2$, there is an n_2' such that $n_2 \xrightarrow{a} n_2'$ and $n_1' \sqsubseteq_{\approx g} n_2'$.

(b) Next assume $n_1 \xrightarrow{\tau} n_1'$. As n_1 is a normal form, $n_1 \Downarrow \varepsilon$. Furthermore $n_2 \stackrel{\varepsilon}{\Longrightarrow} n_2'$ for some n_2' such that $n_1' \sqsubseteq_{\approx g} n_2'$.

(c) i. If $n_1 \Downarrow \tau$ then $n_2 \Downarrow \tau$ by definition of $\sqsubseteq^{\diamond}_{\approx g}$ and as $\downarrow$ and $\Downarrow \tau$ coincide on NST.

ii. Next assume that $n_1 \Downarrow \tau$, $n_2 \Downarrow \tau$ and $n_2 \xrightarrow{\tau} n_2'$. By definition of $\sqsubseteq^{\diamond}_{\approx g}$, $n_1 \xrightarrow{\tau} n_2'$ for some n_1' where $n_1' \sqsubseteq_{\approx g} n_2'$.

(d) i. Assume $n_1 \Downarrow a$. We will prove that $n_2 \Downarrow a$. As $n_1 \Downarrow$ then $n_2 \Downarrow$. So assume $n_1 \Downarrow$, $n_2 \Downarrow$ and $n_1 \Downarrow a$ but that $n_2 \Uparrow a$. This implies that $\langle a, \{\perp\}\rangle$ is an element in n_2 but not in n_1. It is easy to see that this contradicts the fact that $n_1 \sqsubseteq^{\diamond}_{\approx g} n_2$ and that n_1 and n_2 are normal forms.

ii. Next assume that $n_1 \Downarrow a$, $n_2 \Downarrow a$ and $n_2 \xrightarrow{a} n_2'$. Now $n_1 \sqsubseteq^{\diamond}_{\approx g} n_2$, $n_1 \downarrow$ and $n_2 \downarrow$ and hence $n_1 \xrightarrow{a} n_1'$ for some n_1' such that $n_1 \sqsubseteq_{\approx g} n_1'$.

References

[Abr91] S. Abramsky. A domain equation for bisimulation. *Information and Computation*, 92:161–218, 1991.

[AH92] L. Aceto and M. Hennessy. Termination, deadlock and divergence. *Journal of the ACM*, 39(1):147–187, January 1992.

[AJ95] S. Abramsky and A. Jung. Domain theory. In *Handbook of Logic in Computer Science*. Oxford University Press, 1994.

[Gue81] I. Guessarian. *Algebraic Semantics*, volume 99 of *Lecture Notes in Computer Science*. Springer-Verlag, 1981.

[Hen81] M. Hennessy. A term model for synchronous processes. *Information and Computation*, 51(1):58–75, 1981.

[Hen85] M. Hennessy. Acceptance trees. *Journal of the ACM*, 32(4):896–928, 1985.

[Hen88a] M. Hennessy. *Algebraic Theory of Processes*. MIT Press, Cambridge, Massachusetts, 1988.

[Hen88b] M. Hennessy. Axiomatising finite concurrent processes. *SIAM Journal on Computing*, 17(5):997–1017, 1988.

[Kel76] R.M. Keller. Formal verification of parallel programs. *Communications of the ACM*, 19(7):371–384, 1976.

[Mil80] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.

[Mil81] R. Milner. Modal characterisation of observable machine behaviour. In G. Astesiano and C. Bohm, editors, *Proceedings CAAP 81*, volume 112 of *Lecture Notes in Computer Science*, pages 25–34. Springer-Verlag, 1981.

[Mil83] R. Milner. Calculi for synchrony and asynchrony. *Theoretical Computer Science*, 25:267–310, 1983.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall International, Englewood Cliffs, 1989.

[Par81] D.M.R. Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, 5^{th} *GI Conference*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer-Verlag, 1981.

[Plo81] G.D. Plotkin. Lecture notes in domain theory, 1981. University of Edinburgh.

[SS71] D.S. Scott and C. Strachey. Towards a mathematical semantics for computer languages. In *Proceedings of the Symposium on Computers and Automata*, volume 21 of *Microwave Research Institute Symposia Series*, 1971.

[Sti87] C. Stirling. Modal logics for communicating systems. *Theoretical Computer Science*, 49:311–347, 1987.

[Wal90] D.J. Walker. Bisimulation and divergence. *Information and Computation*, 85(2):202–241, 1990.