INFORMATION AND COMPUTATION 103, 204–269 (1993)

Towards Action-Refinement in Process Algebras*

L. ACETO AND M. HENNESSY

School of Cognitive and Computing Sciences, University of Sussex, Falmer, Brighton BN1 9QH, United Kingdom

We present a simple process algebra which supports a form of refinement of an action by a process and address the question of an appropriate equivalence relation for it. The main result of the paper is that an adequate equivalence can be defined in a very intuitive manner. In fact we show that it coincides with the timed-equivalence proposed by one of the authors. We also show that it can be characterized equationally.

Contents

1. INTRODUCTION

The term *reactive systems* has recently been coined [Pn85] to refer to a wide class of computational systems which evolve or compute by periodically interacting with their environment.

These systems may be purely software, such as processes written in the programming language Occam [OC84], purely hardware, such as VLSI systems, or a mixture of both, such as communication protocols. Indeed,

* This work has been supported by a grant from the United Kingdom Science and Engineering Research Council and by the Esprit BRA Project CEDISYS.

0890-5401/93 $5.00

most computational devices may be viewed in one way or another as reactive systems; so that a theory of these systems is also a general theory of concurrent systems.

Many languages have been devised for describing reactive systems; in this paper we restrict our attention to the so-called *process algebras* such as **CCS** [Mil80], **CSP** [Hoare85], and **ACP** [BK85]. These consist of a set of combinators for constructing new systems from existing ones together with a mechanism for recursive definitions and are normally parameterized with respect to a predefined set of actions. For example, if a, b, c are actions then P, defined by

$$P \Leftarrow (a; P) + (b; c; P),$$

describes a reactive system which can either perform an a-action and proceed as before or perform a b-action followed by a c-action and proceed as before.

These languages are designed to describe not only actual systems but also their specifications. It follows that a very important component for these languages is a notion of equivalence between descriptions: one description might be a specification, say SPEC, and another, SYS, the description of an actual implementation and to say that they are equivalent means that SYS is a correct implementation of SPEC, for both describe essentially the same behaviour but at different levels of *abstraction* or *refinement*.

A variety of equivalences have been proposed [Mil80, HM85, Hoare85, DH84], and a variety of methods for proving pairs of descriptions equivalent. Although the resulting theories are very different, speaking in general terms they have much in common. They all allow abstraction from internal actions and, in addition, the equivalences in the latter two references allow descriptions to be equivalent if they both describe the same behaviour modulo *nondeterminism*.

We would like to develop a language for reactive systems and a related equivalence which would support a form of abstraction/refinement where actions might be refined to processes or dually processes might be abstracted to actions. For example, at a very high level of abstraction a system might be described by

$$\text{SPEC} \Leftarrow \text{input}; \text{output}; \text{SPEC}.$$

At a more refined level it may be that the actions of input and output are rather complicated and are carried out by processes P and Q, respectively. So SPEC could be refined to

$$\text{SYS} \Leftarrow P; Q; \text{SYS}.$$

This form of abstraction/refinement is not supported by any existing Process Algebra and it is the topic of the present paper. More specifically we consider a very simple (even minimal) process algebra, and a new combinator for refining an action by a process and address the question of an appropriate equivalence for the augmented language. The main result of the paper is that, at least for the simple language we consider, an adequate equivalence can be defined in a very intuitive manner and moreover can be axiomatised in much the same way as the standard behavioural equivalences [HM85,DH84].

Semantic equivalences for models of concurrent computation, e.g., *Petri Nets* [Rei85] and *Event Structures* [Win87], which support the refinement of actions by processes have recently been the subject of extensive study in the literature; see, e.g., [GG88, Gl90a] for well-written surveys. For example, history preserving bisimulation and ST-bisimulation equivalences over prime Event Structures have been shown to be preseved by action refinement in [GG88, Gl90], respectively. Refinement theorems for branching bisimulation [GW89] over process graphs [BK85] and causal trees [DD89a] have been given in [GW89a, DD89b]. Vogler proves in [Vo90] that a variation on failure semantics based on interval semiwords is a congruence with respect to action refinement over safe Petri Nets. A syntactic study of action refinement in the framework of trace equivalence over a simple process algebra has been presented in [NEL88]. In this paper we shall set the basis for a syntactic study of action refinement in process algebras in the semantic framework of bisimulation equivalence. Further developments of this theory for more complex algebras may be found in [AH90]. We refer the interested reader to Section 6 for a more detailed comparison with related work.

We now give a brief outline of the remainder of the paper. In Section 2 we define our language, which is deliberately designed to be the simplest possible in which the questions we address are interesting. We also give a standard operational semantics in terms of a labelled transition system [Pl81]. This is the starting point for most of the existing behavioural equivalences. We give one example, strong observational equivalence $\sim$, and show that it is not adequate for our language. The basic problem is that two equivalent terms can have different effects when used as part of a larger system or context. It means that this equivalence will not support a compositional proof method. Nevertheless it is an intuitive equivalence and we argue that an appropriate equivalence for our language is the largest context-preserving equivalence contained in $\sim$, $\sim^c$. We show that this relation $\sim^c$ is intuitively and mathematically tractable, the former by showing that it coincides with timed-equivalence [H88] and the latter by showing that it can be equationally characterized.

Timed-equivalence is the subject of Section 3. If an action can be refined

by a process, we can no longer consider actions as being atomic events. A minimal consequence is that actions have distinct beginnings and endings. This is the intuition underlying timed-equivalence, which we denote by $\sim_t$. It is strong observational equivalence but defined using subactions, beginnings and endings, $S(a)$ and $F(a)$ for each action a.

We prove that $\sim_t$ and $\sim^c$ coincide. However, the proof of $\sim_t \subseteq \sim^c$ relies on an equational characterization of $\sim_t$. This equational characterization is given in detail in Section 5 as it is independent of the remainder of the paper. One consequence of the fact that $\sim_t$ and $\sim^c$ coincide is that $\sim_t$ is preserved by action-refinements; i.e., $p \sim_t q$ implies that $p[a \rightsquigarrow r] \sim_t q[a \rightsquigarrow r]$, for each process r. However, the proof is very indirect as it relies on the completeness of a set of equations for $\sim_t$ and the particular form these equations take. It would be preferable to have an operational or behavioural proof of this fact as we are unlikely to have complete equational theories of the appropriate form for more complicated languages.

With this in mind, in Section 4 we introduce a variation of timed-equivalence called *refine-equivalence*, $\sim_r$. This is also a strong observational equivalence defined using the subactions $S(a)$ and $F(a)$, but the extra ingredient is that the beginnings and endings of actions must be properly matched. In many ways this is a more intuitive formulation of strong observational equivalence for non-atomic actions, although the formal definition is somewhat more complicated. We show that, for our simple language, $\sim_t$ and $\sim_r$ coincide and we also give a purely behavioural proof of the fact that $\sim_r$ is preserved by action-refinements.

We end with a conclusion and suggestions for future work. A table of the notations used in the paper may be found in the Appendix.

2. The Language

In this section we present the language used in this paper to discuss the problems mentioned in the introduction. The language we use is a process algebra, in the style of **CCS** [Mil80], **CSP** [Hoare85], and **ACP** [BK85], equipped with a new combinator for refining an action by a process. The set of combinators we consider is a minimal one for the issue we want to address in this paper to be interesting. For instance, it lacks a notion of communication between parallel agents and a facility for defining recursive agents. However, its set of combinators constitutes the core of any process algebra and an understanding of the theory we aim at developing for them has proved to be useful in extensions to more complex algebras which are currently under investigation, [AH90].

This section is organized as follows: in 2.1 we briefly review two standard means of defining the operational semantics of concurrent agents, Labelled Transition Systems (LTS) [Kel76], and bisimulation [Par81]. The notion of (strong) bisimulation is used many times in the remainder of the paper for many different LTS's and we often refer the reader to the general definitions given in this section.

In 2.2 we present our basic language **P** and its operational semantics. The operational rules are used to define a standard observational semantics for **P** by means of the strong bisimulation technique outlined in 2.1.

In 2.3 we enrich **P** with a simple refinement operator $[a \rightsquigarrow q]$. This operator allows the refinement of an action by a process. The operational semantics for this language will be defined in two ways: the first reduces a process $p \in \mathbf{P}_{\text{ref}}$ to a process $\mathbf{red}(p) \in \mathbf{P}$ by essentially considering the new operator as a syntactic substitution. The operational behaviour of p is then indirectly defined by that of $\mathbf{red}(p)$. The second defines a set of transition relations which directly give the operational semantics for $p \in \mathbf{P}_{\text{ref}}$.

The associated strong bisimulations will be shown to coincide on $\mathbf{P}_{\text{ref}}$. We will denote them by $\sim$.

However, as we will show by means of examples, it turns out that $\sim$ is not an adequate semantic equivalence for $\mathbf{P}_{\text{ref}}$. In fact, it is not a congruence with respect to action refinement. This means that $\sim$ would not support compositional proof techniques. As is standard practice we consider the largest congruence, with respect to $\mathbf{P}_{\text{ref}}$, contained in $\sim$. We denote it by $\sim^c$. The remainder of the paper is devoted to studying the properties of $\sim^c$.

2.1. *Labelled Transition Systems and Bisimulation*

A standard way of defining the operational semantics of concurrent processes is the so-called *two-step approach*. This approach, advocated by Milner in [Mil80] and most of his subsequent work, is based on associating an operational semantics to processes following Plotkin's *Structured Operational Semantics* (SOS) [Pl81] and then abstracting from unwanted details on the way processes evolve using a behavioural equivalence which equates processes which cannot be told apart by means of external observation of their behaviour. A semantic process is then taken to be a congruence class of syntactic objects (terms). In this section we briefly review these ideas and refer to the quoted references for more details.

A standard notion which is used in defining the operational semantics of concurrent processes using Plotkin's SOS is that of *Labelled Transition System* (LTS) [Kel76].

Informally an LTS is based on a notion of global state and of transition from one (global) state to another. The transition relation is usually

parameterized on a set of atomic actions **Act** and several interpretations of the relations $\xrightarrow{\alpha}$, $\alpha \in \mathbf{Act}$, are possible. A standard one is the following [Mil80]:

$p \xrightarrow{\alpha} p'$ iff p may perform the action α and become p' in doing so.

Formally the definition of labelled transition system is the following one:

DEFINITION 2.1. A *Labelled Transition System* (LTS) is a triple $(\mathbf{St}, \mathbf{Act}, \{\xrightarrow{\alpha} \mid \alpha \in \mathbf{Act}\})$, where

- **St** is a set of *states*,
- **Act** is a set of *actions*,
- for each $\alpha \in \mathbf{Act}$, $\xrightarrow{\alpha} \subseteq \mathbf{St}^2$ is called a *transition relation*.

The second step of the operational approach to the semantics of concurrent processes is *abstraction from unwanted details*. This is achieved by factorizing the set of processes via one of the observational equivalences proposed in the literature, [Mil80, Mil89, DH84, BHR84]. The essence of these equivalences is that they are based upon the idea of *observing* a process, which is usually taken to mean communicating with it. The idea is that processes are equal iff they are indistinguishable in any experiment based upon observation. One of the most extensively studied approaches to the definition of observational equivalences for processes is the one based on Park's notion of *bisimulation* [Par81]. This way of defining observational equivalences on processes will be frequently used in this paper. For this reason we now give the formal definition of bisimulation for LTS and will refer to it every time we will define an equivalence in this style.

DEFINITION 2.2. Let $(\mathbf{St}, \mathbf{Act}, \{\xrightarrow{\alpha} \mid \alpha \in \mathbf{Act}\})$ be an LTS.
A binary relation $\mathcal{R}$ on **St** is called a *bisimulation* if for each $(s_1, s_2) \in \mathcal{R}$, $\alpha \in \mathbf{Act}$ the following clauses hold:

(i) $s_1 \xrightarrow{\alpha} s_1' \Rightarrow \exists s_2' : s_2 \xrightarrow{\alpha} s_2'$ and $(s_1', s_2') \in \mathcal{R}$,
(ii) $s_2 \xrightarrow{\alpha} s_2' \Rightarrow \exists s_1' : s_1 \xrightarrow{\alpha} s_1'$ and $(s_1', s_2') \in \mathcal{R}$.

The following result is then standard:

PROPOSITION 2.1. *Let* $\sim = \bigcup \{\mathcal{R} \subseteq \mathbf{St}^2 \mid \mathcal{R}$ *is a bisimulation*$\}$. *Then*:

(i) $\sim$ *is the maximum bisimulation.*
(ii) $\sim$ *is an equivalence relation on* **St**.

In the remainder of the paper this technique for defining the operational semantics of concurrent processes is used several times. We will introduce

simple languages $\mathscr{L}$ and will define the operational semantics of the terms by induction upon their structure. This associates an LTS to each term $t \in \mathscr{L}$ and thus to $\mathscr{L}$ itself. Consequently we apply the notion of bisimulation, described above for general LTS's, to obtain a standard observational equivalence.

2.2. *A Basic Language*

The language we consider, which is closely related to the language PA investigated by Bergstra and Klop in several papers in the literature, see, e.g., [BK84, 88], will be parameterized over a set of actions **A**. The set of actions **A** will be a subset of the set of processes and actions, together with the terminated process *nil*, and will be the constants of our algebra of processes. The process combinators used to build new systems from existing ones will be the following:

- $+$ for *non-deterministic choice*,
- $;$ for *sequential composition*,
- $|$ for *parallel composition*, and
- $\lfloor$ for *left-merge* [BK84, H88, CH89].

Formally:

DEFINITION 2.3. Let **A** be an uninterpreted set of *actions*. **A** is ranged over by $a, b, a', b', \ldots$.

For each $n \in N$ let Σ^n be defined as follows:

(i) $\Sigma^0 = \{nil\} \cup \mathbf{A}$,

(ii) $\Sigma^2 = \{+, ;, |, \lfloor\}$,

(iii) $\Sigma^n = \varnothing \ \forall n \notin \{0, 2\}$.

The signature Σ is defined as $\Sigma =_{\text{def}} \bigcup_{n \geq 0} \Sigma^n$. Let **P** denote the word algebra over Σ, $\mathbf{T}_\Sigma$.

Terms in **P** are written considering $+, ;, |$, and $\lfloor$ as infix operators. **P** is ranged over by $p, q, q', q_1, \ldots$, and it is given the structure of a labelled transition system by defining a standard operational semantics for it in Plotkin's SOS style [Mil80, Pl81]. Due to the fact that we are considering general sequential composition rather than action-prefixing, we need a termination predicate on **P** (another alternative is shown in [BHR84]. Here, due to the simplicity of the language we make do with a syntactically defined termination predicate).

DEFINITION 2.4. Let $\surd$ be the least set which satisfies

- $nil \in \surd$

- $q_1 \in \surd \Rightarrow \forall q \in \mathbf{P} q_1 \between q \in \surd$
- $q_1, q_2 \in \surd \Rightarrow q_1 + q_2, q_1 ; q_2, q_1 \mid q_2 \in \surd$.

Notation 2.1. We will write $q\surd$ for $q \in \surd$.

Intuitively, for each $q \in \mathbf{P}$, $q\surd$ iff q is a terminated process, which will mean, as it will become clear after the next definition, that q cannot perform any transition.

DEFINITION 2.5. For each $a \in \mathbf{A}$, $\xrightarrow{a}$ is the least binary relation on $\mathbf{P}$ which satisfies the following axiom and rules:

1. $a \xrightarrow{a} nil$
2. $q_1 \xrightarrow{a} q' \Rightarrow q_1 + q_2 \xrightarrow{a} q', q_2 + q_1 \xrightarrow{a} q'$
3. $q_1 \xrightarrow{a} q' \Rightarrow q_1 ; q_2 \xrightarrow{a} q' ; q_2$
4. $q_1\surd, q_2 \xrightarrow{a} q' \Rightarrow q_1 ; q_2 \xrightarrow{a} q'$
5. $q_1 \xrightarrow{a} q' \Rightarrow q_1 \mid q_2 \xrightarrow{a} q' \mid q_2, q_2 \mid q_1 \xrightarrow{a} q_2 \mid q'$
6. $q_1 \xrightarrow{a} q' \Rightarrow q_1 \between q_2 \xrightarrow{a} q' \mid q_2$.

We may now define a standard observational equivalence on $\mathbf{P}$ using the notion of strong bisimulation presented in the previous section. The equivalence relation generated in this way for the language $\mathbf{P}$ will be denoted by $\sim$.

PROPOSITION 2.2. *$\sim$ is a Σ-congruence.*

It is natural to require that semantic equivalence relations be congruences. This means that whenever we have two equivalent specifications we may place either of them into a larger system (or *context*) obtaining equivalent behaviours.

An important feature of the theory of $\sim$ is that $\mid$ is not a primitive operator, at least for finite processes. In fact, every finite parallel process has an equivalent purely non-deterministic counterpart.

This is illustrated by the following standard example.

EXAMPLE 2.1. Let $q_1 = a \mid b$ and $q_2 = (a ; b) + (b ; a)$. Then

$$\mathcal{R} = \{(q_1, q_2), (a \mid nil, nil ; a), (nil \mid b, nil ; b), (nil \mid nil, nil)\}$$

is a bisimulation. Hence $q_1 \sim q_2$.

Another important feature of strong bisimulation equivalence is that its theory has several complete axiomatizations over different languages [HM85, H88, BK85]. This shows that this equivalence is mathematically

tractable. In what follows we would like to show the same property for a language based on **P** enriched with a feature for action-refinement.

2.3. *The Process Language with Refinement*

In this section we introduce the language whose semantic properties will be investigated in the remainder of this paper. The language, which we call the *process language with refinement*, $\mathbf{P}_{\text{ref}}$, is based on the signature Σ enriched with a combinator, $[a \rightsquigarrow p]$, which allows the refinement of an action by a process p. Formally:

DEFINITION 2.6. The *process language with refinement* $\mathbf{P}_{\text{ref}}$ is the language generated by the grammar

$$p ::= nil \mid a \mid p + p \mid p ; p \mid p \mid p \mid p \between p \mid p[a \rightsquigarrow p],$$

where $a \in \mathbf{A}$.

$\mathbf{P}_{\text{ref}}$ will be ranged over, with abuse of notation, by $p, q, r, p', p_1, \ldots$.

EXAMPLE 2.2. $p = (a \mid b)[a \rightsquigarrow a_s ; a_f]$ stands for a process in which action a has been refined to the sequential composition of two subactions a_s and a_f.

With regard to the simple process presented in the above example, operationally we would expect it to be able to perform any shuffle of the strings $a_s a_f$ and b. We have at least two ways of capturing the operational behaviour of the processes definable in $\mathbf{P}_{\text{ref}}$:

- We might regard $p[a \rightsquigarrow p']$ as indicating the syntactic substitution of process p' for any occurrence of a in p. Iterating this procedure we could translate or reduce every process in $\mathbf{P}_{\text{ref}}$ into a process in **P**. The operational behaviour of $p[a \rightsquigarrow p']$ would then be determined by that of its translation according to the operational semantics of **P**-processes.
- we might explicitly define an operational semantics for $\mathbf{P}_{\text{ref}}$.

We follow both lines and show that the resulting semantic equivalences coincide. Having done so, in the following sections of this paper we use the operational semantics of $\mathbf{P}_{\text{ref}}$-processes given via the reduction to **P**. This method, although it may be less elegant, will be technically more tractable.

DEFINITION 2.7. 1. The *reduction function*, $\mathbf{red} : \mathbf{P}_{\text{ref}} \to \mathbf{P}$, is defined by structural induction as follows:

(i) $\mathbf{red}(nil) = nil$

(ii) $\mathbf{red}(a) = a$

(iii) $\mathbf{red}(p_1 \text{ op } p_2) = \mathbf{red}(p_1) \text{ op } \mathbf{red}(p_2)$, $\text{op} \in \{+, ;, |, \between\}$

(iv) $\mathbf{red}(p_1[a \rightsquigarrow p_2]) = \mathbf{red}(p_1)[a/\mathbf{red}(p_2)]$, where $[a/\cdot]$ denotes the syntactic substitution of $\mathbf{red}(p_2)$ for each occurrence of a in $\mathbf{red}(p_1)$.

2. A *refinement function* $\rho: \mathbf{A} \to \mathbf{P}_{\text{ref}}$ may be used to generalize the construct $[a \rightsquigarrow p_1]$. The language obtained in this way is denoted by $\mathbf{P}_\rho$. The reduction function for this language is defined as in 1 with the following clause in place of clause (iv),

$$\mathbf{red}(p[\rho]) = \mathbf{red}(p)\,\rho_r$$

where $\rho_r: \mathbf{A} \to \mathbf{P}$ is the syntactic substitution defined by

$$\forall a \in \mathbf{A} \qquad \rho_r(a) = \mathbf{red}(\rho(a)).$$

Intuitively, a refinement function ρ may be seen as the simultaneous refinement of each action $a \in \mathbf{A}$ by the process $\rho(a)$. It is interesting to note that it is essential to consider refinement functions ρ whose codomain is $\mathbf{P}_{\text{ref}}$ rather than $\mathbf{P}_\rho$. In fact, had we allowed refinement maps $\rho: \mathbf{A} \to \mathbf{P}_\rho$, the reduction function $\mathbf{red}(\cdot)$ would not be properly defined for refinements such that $\rho(a) = a[\rho]$.

Fact 2.1. (i) $\forall p \in \mathbf{P}_{\text{ref}}(\mathbf{P}_\rho)\ \mathbf{red}(p) \in \mathbf{P}$.

(ii) $\forall q \in \mathbf{P}\ \text{red}(q) = q$.

With the above notion of reduction each process $p \in \mathbf{P}_{\text{ref}}(\mathbf{P}_\rho)$ inherits an operational semantics from its reduction. For example,

$$\mathbf{red}((a \mid b)[a \rightsquigarrow a_s; a_f]) = (a_s; a_f) \mid b,$$

and the transitions of such a $\mathbf{P}$-process can be easily determined using the operational rules given in the previous section.

Moreover, $\mathbf{P}_{\text{ref}}(\mathbf{P}_\rho)$ inherits a notion of observational equivalence for processes from $\mathbf{P}$ via $\mathbf{red}(\cdot)$. Two processes $p_1, p_2 \in \mathbf{P}_{\text{ref}}$ ($p_1, p_2 \in \mathbf{P}_\rho$) are equivalent iff their reductions are equivalent with respect to $\sim$. From now on we concentrate on $\mathbf{P}_\rho$ as $\mathbf{P}_{\text{ref}}$ is a "sublanguage" of $\mathbf{P}_\rho$.

DEFINITION 2.8. $\forall p_1, p_2 \in \mathbf{P}_\rho\ p_1 \sim_1 p_2 \Leftrightarrow \mathbf{red}(p_1) \sim \mathbf{red}(p_2)$.

We now define an explicit operational semantics for $\mathbf{P}_\rho$ from which a standard observational equivalence is obtained.

First of all we define the termination predicate $\surd$ for the extended language.

DEFINITION 2.9. (i) For each $p \in \mathbf{P}_\rho$ define $\mathbf{Ter}(p) \subseteq \mathbf{A}$, the set of actions of p which have to be mapped to terminated processes to force p to terminate, by structural induction as follows:

- $\mathbf{Ter}(nil) = \varnothing$
- $\mathbf{Ter}(a) = \{a\}$
- $\mathbf{Ter}(p_1 \between p_2) = \mathbf{Ter}(p_1)$
- $\mathbf{Ter}(p_1 \text{ op } p_2) = \mathbf{Ter}(p_1) \cup \mathbf{Ter}(p_2)$, for $\text{op} \in \{+, ;, |\}$
- $\mathbf{Ter}(p[\rho]) = \bigcup_{a \in \mathbf{Ter}(p)} \mathbf{Ter}(\rho(a))$.

(ii) Let $\surd$ be the least subset of $\mathbf{P}_\rho$ which satisfies

- $nil \in \surd$
- $p_1 \in \surd \Rightarrow p_1 \between p_2 \in \surd$
- $p_1, p_2 \in \surd \Rightarrow p_1 \text{ op } p_2 \in \surd$, where $\text{op} \in \{+, ;, |\}$
- $\forall a \in \mathbf{Ter}(p)\ \rho(a) \in \surd \Rightarrow p[\rho] \in \surd$.

In what follows we write $p\surd$ iff $p \in \surd$.

Notation 2.2. Given two refinement functions $\rho, \rho' : \mathbf{A} \to \mathbf{P}_{\text{ref}}$, we write $\rho \circ \rho'$ for the refinement function defined as

$$\forall a \in \mathbf{A}\ (\rho \circ \rho')(a) = \rho'(a)\rho,$$

the application of the syntactic substitution ρ to each $\rho'(a)$.

One can prove by induction on p that $\mathbf{red}(p\rho) = \mathbf{red}(p)\rho_r$, from which it follows that $(\rho \circ \rho')_r = \rho_r \circ \rho'_r$.

We can now define an explicit operational semantics for the language $\mathbf{P}_\rho$ (and hence for its sublanguage $\mathbf{P}_{\text{ref}}$).

DEFINITION 2.10. For each $a \in \mathbf{A}$ let $\xrightarrow{a}$ be the least binary relation on $\mathbf{P}_\rho$ which satisfies the following axiom and rules:

1. $a \xrightarrow{a} nil$
2. $p_1 \xrightarrow{a} p_1' \Rightarrow p_1 + p_2 \xrightarrow{a} p_1',\ p_2 + p_1 \xrightarrow{a} p_1'$
3. $p_1 \xrightarrow{a} p_1' \Rightarrow p_1 ; p_2 \xrightarrow{a} p_1' ; p_2$
4. $p_1\surd,\ p_2 \xrightarrow{a} p' \Rightarrow p_1 ; p_2 \xrightarrow{a} p'$
5. $p_1 \xrightarrow{a} p_1' \Rightarrow p_1 \mid p_2 \xrightarrow{a} p_1' \mid p_2,\ p_2 \mid p_1 \xrightarrow{a} p_2 \mid p_1'$
6. $p_1 \xrightarrow{a} p_1' \Rightarrow p_1 \between p_2 \xrightarrow{a} p_1' \mid p_2$
7. $\rho(b) \xrightarrow{a} p' \Rightarrow b[\rho] \xrightarrow{a} p'$
8. $p_1[\rho] \xrightarrow{a} p' \Rightarrow (p_1 + p_2)[\rho] \xrightarrow{a} p',\ (p_2 + p_1)[\rho] \xrightarrow{a} p'$
9. $p_1[\rho] \xrightarrow{a} p' \Rightarrow (p_1 ; p_2)[\rho] \xrightarrow{a} p' ; (p_2[\rho])$
10. $p_1[\rho]\surd,\ p_2[\rho] \xrightarrow{a} p' \Rightarrow (p_1 ; p_2)[\rho] \xrightarrow{a} p'$
11. $p_1[\rho] \xrightarrow{a} p_1' \Rightarrow (p_1 \mid p_2)[\rho] \xrightarrow{a} p_1' \mid (p_2[\rho]),\ (p_2 \mid p_1)[\rho] \xrightarrow{a} (p_2[\rho]) \mid p_1'$

12. $p_1[\rho] \xrightarrow{a} p_1' \Rightarrow (p_1 \curlyvee p_2)[\rho] \xrightarrow{a} p_1' \mid (p_2[\rho])$
13. $p[\rho' \circ \rho] \xrightarrow{a} p' \Rightarrow (p[\rho])[\rho'] \xrightarrow{a} p'$.

Using this operational semantics we can define a standard observational equivalence on $\mathbf{P}_\rho$ using the strong bisimulation technique as we have done for $\mathbf{P}$. The resulting equivalence relation on $\mathbf{P}_\rho$ is denoted by $\sim_2$. We now show that $\sim_1$ and $\sim_2$ coincide on $\mathbf{P}_\rho$. To do so we need to relate the moves of a process $p \in \mathbf{P}_\rho$ with those of its reduction $\mathbf{red}(p) \in \mathbf{P}$.

LEMMA 2.1. *The following statements hold*:

(i) $\forall p \in \mathbf{P}_\rho\ \mathbf{Ter}(p) = \mathbf{Ter}(\mathbf{red}(p))$.
(ii) $\forall p \in \mathbf{P}_\rho\ p\surd \Leftrightarrow \mathbf{red}(p)\surd$.

Proof. Both the statements can be shown by induction on the structure of p. The only interesting case is $p = p_1[\rho]$.

(i) $\mathbf{Ter}(p_1[\rho]) = \bigcup_{a \in \mathbf{Ter}(p_1)} \mathbf{Ter}(\rho(a))$, by definition.
This is equal to $\bigcup_{a \in \mathbf{Ter}(\mathbf{red}(p_1))} \mathbf{Ter}(\mathbf{red}(\rho(a)))$, by the inductive hypothesis, which by definition is equal to $\mathbf{Ter}(\mathbf{red}(p_1)\rho_r)$.

(ii) $p_1[\rho]\surd \Leftrightarrow \forall a \in \mathbf{Ter}(p_1)\ \rho(a)\surd$, by definition.
By inductive hypothesis, this is equivalent to

$$\forall a \in \mathbf{Ter}(p_1)\ \mathbf{red}(\rho(a))\surd.$$

By clause (i) of the lemma, this is equivalent to

$$\forall a \in \mathbf{Ter}(\mathbf{red}(p_1)) \qquad \mathbf{red}(\rho(a))\surd.$$

This is in turn equivalent to $(\mathbf{red}(p_1)\rho_r)\surd$ and $\mathbf{red}(p_1[\rho])\surd$.

This completes the proof of the lemma. ∎

We can now relate the moves of a process $p \in \mathbf{P}_\rho$ to those of its reduction $\mathbf{red}(p)$.

LEMMA 2.2. *For each* $p \in \mathbf{P}_\rho$ *the following statements hold*:

(i) $p \xrightarrow{a} p' \Rightarrow \mathbf{red}(p) \xrightarrow{a} \mathbf{red}(p')$;
(ii) $\mathbf{red}(p) \xrightarrow{a} x \Rightarrow \exists p' : p \xrightarrow{a} p'$ *and* $\mathbf{red}(p') = x$.

Proof. We prove both statements by induction on the length n of the proof of $p \xrightarrow{a} p'$ and $\mathbf{red}(p) \xrightarrow{a} x$, respectively.

We proceed by examining the structure of p and sketch the details of two cases of the proof of statement (i) leaving the others to the reader.

$p = p_1; p_2$. Assume $p_1; p_2 \xrightarrow{a} p'$. There are two subcases to examine:

(a) $p_1 \xrightarrow{a} p_1'$ and $p' = p_1'; p_2$.

The proof of the derivation $p_1 \xrightarrow{a}_{\cdot} p_1'$ has length $n-1$. Hence, by inductive hypothesis, $\mathbf{red}(p_1) \xrightarrow{a} \mathbf{red}(p_1')$.

By the operational semantics for $\mathbf{P}$,

$$\mathbf{red}(p_1; p_2) = \mathbf{red}(p_1); \mathbf{red}(p_2) \xrightarrow{a} \mathbf{red}(p_1'); \mathbf{red}(p_2) = \mathbf{red}(p_1'; p_2).$$

(b) $p_1\surd$ and $p_2 \xrightarrow{a}_{\cdot} p'$.

The proof of the derivation $p_2 \xrightarrow{a} p'$ has length $n-1$. Hence, by inductive hypothesis, $\mathbf{red}(p_2) \xrightarrow{a} \mathbf{red}(p')$.

By the above lemma, $p_1\surd \Leftrightarrow \mathbf{red}(p_1)\surd\mathbf{P}$.

Hence, by the operational semantics for $\mathbf{P}$,

$$\mathbf{red}(p_1; p_2) = \mathbf{red}(p_1); \mathbf{red}(p_2) \xrightarrow{a} \mathbf{red}(p').$$

$p = p_1[\rho]$. We now examine the structure of p_1 concentrating on showing the claim for two subcases.

$p_1 = b$. Then $\rho(b) \xrightarrow{a}_{\cdot} p'$. The proof of this derivation has length $n-1$. Hence, by inductive hypothesis, $\mathbf{red}(\rho(b)) \xrightarrow{a} \mathbf{red}(p')$.

This implies $\mathbf{red}(b[\rho]) = b\rho_r \xrightarrow{a} \mathbf{red}(p')$.

$p_1 = p_2[\rho']$. Assume $(p_2[\rho'])[\rho] \xrightarrow{a}_{\cdot} p'$.

Then $p_2[\rho \circ \rho'] \xrightarrow{a}_{\cdot} p'$.

The length of the proof of this derivation is less than n. Hence, by inductive hypothesis, $\mathbf{red}(p_2[\rho \circ \rho']) \xrightarrow{a} \mathbf{red}(p')$.

Moreover, by definition of $\mathbf{red}(\cdot)$,

$$\mathbf{red}(p_2[\rho \circ \rho']) = \mathbf{red}(p_2)(\rho \circ \rho')_r.$$

By a previous observation (Notation 2.2), $\mathbf{red}(p_2)(\rho \circ \rho')_r = \mathbf{red}(p_2)(\rho_r \circ \rho_r')$.

It follows that

$$\mathbf{red}(p_2)(\rho_r \circ \rho_r') = (\mathbf{red}(p_2)\,\rho_r')\,\rho_r = \mathbf{red}((p_2[\rho'])[\rho]).$$

This completes the inductive argument for statement (i). The proof of statement (ii) follows entirely similar lines. ∎

We now show that the equivalences $\sim_1$ and $\sim_2$ coincide on $\mathbf{P}_\rho$ (and thus on $\mathbf{P}_{\mathrm{ref}}$). This is stated by the following theorem.

THEOREM 2.1. $\forall p_1, p_2 \in \mathbf{P}_\rho\; p_1 \sim_1 p_2 \Leftrightarrow p_1 \sim_2 p_2$.

Proof. We will show that for each $p_1, p_2 \in \mathbf{P}_\rho$,

$$p_1 \sim_2 p_2 \Leftrightarrow \mathbf{red}(p_1) \sim \mathbf{red}(p_2).$$

(*Only if*) Consider the relation $\mathscr{R} \subseteq \mathbf{P}^2$ defined as follows:

$$\mathscr{R} \underset{\text{def}}{=} \{(\mathbf{red}(p_1), \mathbf{red}(p_2)) \mid p_1 \sim_2 p_2\}.$$

We show that $\mathscr{R}$ is a **P**-bisimulation. By symmetry it is sufficient to show that

$$\mathbf{red}(p_1) \xrightarrow{a} x \Rightarrow \exists y : \mathbf{red}(p_2) \xrightarrow{a} y \text{ and } (x, y) \in \mathscr{R}.$$

Assume $\mathbf{red}(p_1) \xrightarrow{a} x$. Then, by the above lemma,

$$\exists p_1' : p_1 \xrightarrow{a} p_1' \text{ and } \mathbf{red}(p_1') = x.$$

As $p_1 \sim_2 p_2$, there exists p_2' such that

$$p_2 \xrightarrow{a} p_2' \text{ and } p_1' \sim_2 p_2'.$$

By the above lemma, $\mathbf{red}(p_2) \xrightarrow{a} \mathbf{red}(p_2')$ and, by definition of $\mathscr{R}$, $(\mathbf{red}(p_1'), \mathbf{red}(p_2')) \in \mathscr{R}$. Hence $\mathscr{R}$ is a **P**-bisimulation.

(*If*) Consider the relation defined as follows:

$$\mathscr{R} \underset{\text{def}}{=} \{(p_1, p_2) \mid (\mathbf{red}(p_1), \mathbf{red}(p_2)) \in \sim\}.$$

$\mathscr{R}$ can be shown to be a $\mathbf{P}_{\text{ref}}$-bisimulation using the same approach as in the *only if* case. ∎

This equivalence over $\mathbf{P}_{\text{ref}}$, $\sim_1 = \sim_2$, is a conservative extension of $\sim$ over **P** and, for convenience, in future we also use $\sim$ to denote it. We usually use its characterizations in terms of the function $\mathbf{red}(\cdot)$.

However, $\sim$ is not an adequate semantic equivalence for $\mathbf{P}_{\text{ref}}$; it turns out not to be a congruence with respect to the combinator of action-refinement, as the following example shows.

EXAMPLE 2.3. 1. Consider $p = a \mid b$ and $q = (a; b) + (b; a)$. As already noted $p \sim q$. However,

$$p[a \rightsquigarrow a_s; a_f] \not\sim q[a \rightsquigarrow a_s; a_f].$$

In fact,

$$\mathbf{red}(q[a \rightsquigarrow a_s; a_f]) = ((a_s; a_f); b) + (b; (a_s; a_f)) \xrightarrow{a_s} (nil; a_f); b,$$

a state in which only action a_f is possible. No such state can be reached by $\mathbf{red}(p[a \rightsquigarrow a_s; a_f])$ via an action a_s.

2. Consider $q' = (a \mid b) + (a; b)$. Then $p \sim q'$ but

$$p[a \rightsquigarrow a_s; a_f] \not\sim q'[a \rightsquigarrow a_s; a_f].$$

In fact,

$$\mathbf{red}(q'[a \rightsquigarrow a_s; a_f]) = ((a_s; a_f) \mid b) + ((a_s; a_f); b) \xrightarrow{a_s} (nil; a_f); b,$$

a state in which only action a_f is possible.

We have already seen how no equivalent state can be reached by $\mathbf{red}(p[a \rightsquigarrow a_s; a_f])$ via an a_s-action.

As $\sim$ is not a congruence with respect to action-refinement, it would not support compositional proof methods on $\mathbf{P}_{ref}$. However, bisimulation equivalence is a natural notion of observational equivalence between processes (as argued for example in [Mil89, HM85, Ab87]) and we would like to base our theory of action-refinement or a similar notion of equivalence.

We have a standard way of associating a congruence with $\sim$. It is sufficient to close $\sim$ with respect to all $\mathbf{P}_{ref}$-contexts, [Mil80]. The resulting congruence, which we denote by $\sim^c$, is known to be the largest congruence contained in $\sim$, [Mil80]. The definition of $\sim^c$ is, however, purely algebraic and does not shed much light on its behavioural significance. Moreover, it does not support useful proof techniques to show that two processes are related with respect to it. The remainder of the paper is devoted to addressing these two issues.

3. Timed-Equivalence

It has been pointed out in the introduction that, if we want to describe the behaviour of concurrent processes to allow for a step-wise refinement of the actions they perform, the view of processes enforced by interleaving-style equivalences becomes inadequate.

Essentially this depends on the fact that an interleaving view of the behaviour of the processes under consideration strongly depends on what are regarded to be the atomic actions processes may perform.

As in this development we allow for an operation which changes the level of atomicity of actions without enforcing any mutual exclusion policy, we need a more refined behavioural description of the processes than the ones given by the interleaving based equivalences proposed in the literature [HM85, DH84, BHR84].

In [H88] one of the authors has proposed an alternative version of bisimulation equivalence based on actions which are not necessarily instantaneous. A comprehensive description of the resulting semantic theory is given in the above quoted reference. For our purposes it will be sufficient to remind the reader that the resulting behavioural description of processes is based on the assumption that there are observers which can detect the beginning and ending of actions.

A standard bisimulation equivalence can be developed using the operational description of the behaviour of the processes. The result is a semantic theory of processes which distinguishes concurrency from nondeterminism and which can be completely axiomatized [H88].

In this section we develop a semantic theory based on these ideas for the language **P** presented in Section 2. **P** is an extension of the language used in [H88] as action prefixing is replaced by sequential composition of processes. The theory will be developed according to the standard two-step operational description of processes. First of all we define an operational semantics for processes based on the ideas in [H88]. Second we abstract from unwanted details by means of a semantic equivalence based on this operational semantics. This section ends with a discussion of the relevance of this notion of equivalence for processes with respect to the step-wise refinement of actions proposed in Section 2.

3.1. *Timed-Operational Semantics*

We assume that beginnings and terminations of actions are distinct subactions which may be observed. For each $a \in \mathbf{A}$ we use $S(a)$ and $F(a)$ to denote the beginning and the termination of an a-action, respectively. We view $S(a)$, $F(a)$ as a new class of actions and define an operational semantics in terms of new next-event relations $\xrightarrow{S(a)}$, $\xrightarrow{F(a)}$.

Notation 3.1. $\mathbf{A}_s =_{\text{def}} \{S(a) \mid a \in \mathbf{A}\} \cup \{F(a) \mid a \in \mathbf{A}\}$ is called the set of subactions and is ranged over by $e, e', e_1, \ldots$.

As pointed out in [H88], the language for processes proposed in Section 2 is not sufficiently expressive to describe all possible states a process may reach. To overcome this problem we introduce into the language a new symbol $F(a)$ for each $a \in \mathbf{A}$. $F(a)$ will denote the state in which the atomic process a is being executed but has not yet terminated its execution.

DEFINITION 3.1 (Process States). Let $\mathcal{S}$, the set of *process states*, be the least set which satisfies

- $p \in \mathbf{P}$ implies $p \in \mathcal{S}$,
- $a \in \mathbf{A}$ implies $F(a) \in \mathcal{S}$,
- $s \in \mathcal{S}$, $p \in \mathbf{P}$ implies $s; p \in \mathcal{S}$,
- $s_1 \in \mathcal{S}$, $s_2 \in \mathcal{S}$ implies $s_1 \mid s_2 \in \mathcal{S}$.

$\mathcal{S}$ is ranged over by $s, s_1, s', \ldots$.

We can now define a standard operational semantics for states following the pattern described in Section 2. In defining the operational semantics we

use the termination predicate defined in Section 2. This predicate can be easily extended to $\mathscr{S}$ as follows:

DEFINITION 3.2. $\forall s \in \mathscr{S}\ s\surd \Leftrightarrow s \in \mathbf{P}$ and $s\surd$.

DEFINITION 3.3. For each $e \in \mathbf{A}_s$, $\overset{e}{\Longrightarrow}$ is the least binary relation on $\mathscr{S}$ which satisfies the following axioms and rules:

1. $a \overset{S(a)}{\Longrightarrow} F(a)$
2. $F(a) \overset{F(a)}{\Longrightarrow} nil$
3. $p_1 \overset{e}{\Longrightarrow} s_1'$ implies $p_1 + p_2 \overset{e}{\Longrightarrow} s_1'$, $p_2 + p_1 \overset{e}{\Longrightarrow} s_1'$
4. $s_1 \overset{e}{\Longrightarrow} s_1'$ implies $s_1 ; p \overset{e}{\Longrightarrow} s_1' ; p$
5. $s_1\surd$, $p \overset{e}{\Longrightarrow} s$ implies $s_1 ; p \overset{e}{\Longrightarrow} s$
6. $s_1 \overset{e}{\Longrightarrow} s_1'$ implies $s_1 \mid s_2 \overset{e}{\Longrightarrow} s_1' \mid s_2, s_2 \mid s_1 \overset{e}{\Longrightarrow} s_2 \mid s_1'$
7. $p_1 \overset{e}{\Longrightarrow} s$ implies $p_1 \between p_2 \overset{e}{\Longrightarrow} s \mid p_2$.

Rules 1 and 2 above are the new rules which make explicit our view of processes. They state that an atomic process a may perform the beginning of the action a and enter state $F(a)$. Moreover when in state $F(a)$ it can only perform the termination of the action it has started.

A standard behavioural equivalence may now be defined using the above defined operational semantics and the notion of bisimulation given, for arbitrary labelled transtion systems, in Section 2.

DEFINITION 3.4 (Timed Equivalence). The maximum bisimulation over the labelled transition system $\langle \mathscr{S}, \mathbf{A}_s, \{\overset{e}{\Longrightarrow} \mid e \in \mathbf{A}_s\}\rangle$ is denoted by $\sim_t$ and is called *timed (observational) equivalence.*

Some examples of equivalent and inequivalent processes, which may be readily translated into our language, are given in [H88].

We now concentrate on the properties of $\sim_t$ which are relevant to the developments presented in the remainder of the paper. First of all we show that $\sim_t$ is contained in $\sim$ for processes. To do so we need to relate the operational semantics for **P**-processes given in Section 2 with the one given in the above definition.

LEMMA 3.1. *For each* $p \in \mathbf{P}$, $a \in \mathbf{A}$, $p \overset{S(a)}{\Longrightarrow} s \overset{F(a)}{\Longrightarrow} p'$ *iff* $p \overset{a}{\longrightarrow} p'$.

Proof. Both directions of the if-and-only-if can be easily shown by structural induction on p. Note that if $p \overset{S(a)}{\Longrightarrow} s$ then s contains only one occurrence of $F(a)$. With this in mind the proof is straightforward and thus omitted. ∎

We can now show the promised theorem.

THEOREM 3.1. $\forall p, q \in \mathbf{P}\ p \sim_t q \Rightarrow p \sim q$.

Proof. Consider the relation $\mathscr{R} =_{\text{def}} \sim_t \cap \mathbf{P}^2$.

We show that $\mathscr{R}$ is a bisimulation. Take $(p, q) \in \mathscr{R}$. By symmetry, it is sufficient to show that for each $a \in \mathbf{A}$

$$p \xrightarrow{a} p' \text{ implies } \exists q' : q \xrightarrow{a} q' \text{ and } (p', q') \in \mathscr{R}.$$

Assume $p \xrightarrow{a} p'$. Then, by the above lemma, $p \overset{S(a)}{\Longrightarrow} s \overset{F(a)}{\Longrightarrow} p'$ for some $s \in \mathscr{S}$. As $p \sim_t q$, there exists $s' \in \mathscr{S}$ such that $s \sim_t s'$ and $q \overset{S(a)}{\Longrightarrow} s' \overset{F(a)}{\Longrightarrow} q'$ with $p' \sim_t q'$.

By the above lemma, $q \xrightarrow{a} q'$ and, by definition of $\mathscr{R}$, $(p', q') \in \mathscr{R}$.

Hence $\mathscr{R}$ is a bisimulation. This proves the claim. ∎

The reverse implication does not hold as shown by the following example.

EXAMPLE 3.1. Consider $p = (a; b) + (b; a)$ and $q = a \mid b$. Then $p \sim q$ but $p \not\sim_t q$.

In fact, $p \overset{S(a)}{\Longrightarrow} F(a); b$ which is a state in which p can only perform the termination of the action a. No such state can be reached by q via $S(a)$. In fact, $q \overset{S(a)}{\Longrightarrow} s$ implies $s = F(a) \mid b$ and s may perform the start of action b.

PROPOSITION 3.1. $\sim_t$ *is a congruence on* $\mathbf{P}$.

Proof. Standard and thus omitted. ∎

It turns out that $\sim_t$ can be completely axiomatized for the language $\mathbf{P}$ presented in Section 2. The required equations are collected in Fig. 1.

Let $=_E$ be the least $\mathbf{P}$-congruence which satisfies the axioms in Fig. 1.

Then we can state the following result.

THEOREM 3.2. $\forall p, q \in \mathbf{P}\ p =_E q \Leftrightarrow p \sim_t q$.

The proof of this result is rather technical and involved; it is based on the techniques used in, e.g., [CH89, H88] to axiomatize non-interleaving equivalences, but the details are more delicate due to the presence of general sequential composition in the language. Section 5 is entirely devoted to the exposition of the proof. As it is independent of the rest of the paper, we assume the theorem in the remainder of Section 3 and in Section 4.

3.2. $\sim^c$ *Coincides with* $\sim_t$

The remainder of this paper is entirely devoted to proving that $\sim_t$ coincides with $\sim^c$ on the set of processes $\mathbf{P}$. Theorem 3.2 then also provides a

A1 $(x + y) + z = x + (y + z)$

A2 $x + y = y + x$

A3 $x + x = x$

A4 $x + nil = x$

B1 $(x + y) \between z = x \between z + y \between z$

B2 $(x \between y) \between z = x \between (y \mid z)$

B3 $x \between nil = x$

B4 $nil \between x = nil$

X1 $x \mid y = x \between y + y \between x$

C1 $(x; y); z = x; (y; z)$

C2 $x; nil = x = nil; x$

C3 $(x + y); z = (x; z) + (y; z)$ if $x, y \notin \surd$.

FIG. 1. Complete axioms for $\sim_t$.

complete equational characterization of $\sim^c$. As a first step towards proving this claim, in this section we show that $\sim^c$ is contained in $\sim_t$. To show this result we need to *translate states into the language of observers* as stated in [H88]. This is stated formally in the following definition:

DEFINITION 3.5. (i) Let Σ' be the signature defined as follows:

- $\Sigma'_0 = \{S(a) \mid a \in \mathbf{A}\} \cup \{F(a) \mid a \in \mathbf{A}\} \cup \{nil\}$
- $\Sigma'_2 = \{+, ;, \mid, \between\}$
- $\Sigma'_n = \varnothing \ \forall n \neq 0, 2.$

Finally, $\Sigma' = \bigcup_{n \geq 0} \Sigma'_n$.

(ii) The function $\mathbf{Tr}: \mathscr{S} \to \mathbf{T}_{\Sigma'}$ is the unique homomorphism between the two algebras such that:

- $\mathbf{Tr}(nil) = nil$
- $\mathbf{Tr}(a) = S(a); F(a)$
- $\mathbf{Tr}(F(a)) = nil; F(a)$.

The terms in $\mathbf{T}_{\Sigma'}$ may be endowed with an operational description of their behaviour by defining an interleaving-style operational semantics for $\mathbf{T}_{\Sigma'}$. This is done by associating a next-event relation $\xrightarrow{e}_i$ to each $e \in \mathbf{A}_s$ following the definition of $\xrightarrow{a}$, $a \in \mathbf{A}$, given in Section 2.

The relationships between the relations $\stackrel{e}{\Longrightarrow}$ and $\stackrel{e}{\longrightarrow}_i$ are expressed by the following lemma.

LEMMA 3.2. *For each* $s \in \mathcal{S}$, $e \in \mathbf{A}_s$ *the following statements hold*:

1. $s\surd \Leftrightarrow \mathbf{Tr}(s)\surd$;
2. $s \stackrel{e}{\Longrightarrow} s' \Rightarrow \mathbf{Tr}(s) \stackrel{e}{\longrightarrow}_i \mathbf{Tr}(s')$;
3. $\mathbf{Tr}(s) \stackrel{e}{\longrightarrow}_i x \Rightarrow \exists s' : s \stackrel{e}{\Longrightarrow} s'$ *and* $\mathbf{Tr}(s') = x$.

Proof. All the statements can be shown by structural induction on s. We focus on the case $s = s_1 ; p$ for statement 2.

2. Assume $s_1 ; p \stackrel{e}{\Longrightarrow} s'$. There are two cases to consider:

(a) $s_1 \stackrel{e}{\Longrightarrow} s_1'$ and $s' = s_1' ; p$. By inductive hypothesis, $\mathbf{Tr}(s_1) \stackrel{e}{\longrightarrow}_i \mathbf{Tr}(s_1')$. Hence $\mathbf{Tr}(s_1 ; p) = \mathbf{Tr}(s_1); \mathbf{Tr}(p) \stackrel{e}{\longrightarrow}_i \mathbf{Tr}(s_1'); \mathbf{Tr}(p) = \mathbf{Tr}(s_1 ; p)$.

(b) $s_1\surd$ and $p \stackrel{e}{\Longrightarrow} s'$. By inductive hypothesis, $\mathbf{Tr}(p) \stackrel{e}{\longrightarrow}_i \mathbf{Tr}(s')$. By statement 1 of the lemma, $\mathbf{Tr}(s_1)\surd$. Hence $\mathbf{Tr}(s_1); \mathbf{Tr}(p) \stackrel{e}{\longrightarrow}_i \mathbf{Tr}(s')$. ∎

We may define a bisimulation equivalence on $\mathbf{T}_{\Sigma'}$ in the standard way. The details are left to the reader. Let us denote $\sim$, with abuse of notation, the resulting equivalence relation. We concentrate on investigating the relationships between $\sim_t$ and $\sim$ via $\mathbf{Tr}(\cdot)$.

THEOREM 3.3. $\forall s_1, s_2 \in \mathcal{S}\ s_1 \sim_t s_2 \Leftrightarrow \mathbf{Tr}(s_1) \sim \mathbf{Tr}(s_2)$.

Proof. **(Only if)** Assume $s_1 \sim_t s_2$. Define the following relation on $\mathbf{T}_{\Sigma'}$:

$$\mathcal{R} = \{(\mathbf{Tr}(s), \mathbf{Tr}(s')) \mid s \sim_t s'\}.$$

We show that $\mathcal{R}$ is a bisimulation. By symmetry, it is sufficient to show that, for each $(\mathbf{Tr}(s), \mathbf{Tr}(s')) \in \mathcal{R}$, for each $e \in \mathbf{A}_s$

$$\mathbf{Tr}(s) \stackrel{e}{\longrightarrow}_i x \Rightarrow \exists y : \mathbf{Tr}(s') \stackrel{e}{\longrightarrow}_i y \text{ and } (x, y) \in \mathcal{R}.$$

Assume $\mathbf{Tr}(s) \stackrel{e}{\longrightarrow}_i x$. Then, by the above lemma, there exists $\bar{s}$ such that $s \stackrel{e}{\Longrightarrow} \bar{s}$ and $\mathbf{Tr}(\bar{s}) = x$. As $s \sim_t s'$, there exists $\bar{s}'$ such that $s' \stackrel{e}{\Longrightarrow} \bar{s}'$ and $\bar{s} \sim_t \bar{s}'$.

By the above lemma, $\mathbf{Tr}(s') \stackrel{e}{\longrightarrow}_i \mathbf{Tr}(\bar{s}')$ and by definition of $\mathcal{R}$ we have that $(\mathbf{Tr}(\bar{s}), \mathbf{Tr}(\bar{s}')) \in \mathcal{R}$.

(If) Assume $\mathbf{Tr}(s_1) \sim \mathbf{Tr}(s_2)$. Define the following relation on $\mathcal{S}$:

$$\mathcal{R} = \{(s, s') \mid \mathbf{Tr}(s) \sim \mathbf{Tr}(s')\}.$$

Reasoning as above it is easy to show that $\mathcal{R}$ is a timed-bisimulation.

This proves the claim. ∎

COROLLARY 3.1. $\forall p, q \in \mathbf{P}\, p \sim^c q \Rightarrow p \sim_t q$.

Proof. Let ρ_t be the syntactic substitution defined, for all $a \in \mathbf{A}$, by

$$\rho_t(a) = S(a); F(a) \qquad \text{and} \qquad \rho_t(F(a)) = nil; F(a).$$

Note that, for each $s \in \mathscr{S}$, $\mathbf{Tr}(s) = s\rho_t$. Assume now that $p \sim^c q$. Then, for each substitution ρ, $p\rho \sim q\rho$. This implies $p\rho_t \sim q\rho_t$. By the above observation, $p\rho_t = \mathbf{Tr}(p)$ and $q\rho_t = \mathbf{Tr}(q)$. By Theorem 3.3, $\mathbf{Tr}(p) \sim \mathbf{Tr}(q) \Leftrightarrow p \sim_t q$. Hence $p \sim^c q$ implies $p \sim_t q$. ∎

To complete the proof of our claim that $\sim_t$ coincides with $\sim^c$ for the language that we have introduced, we need to show that $\sim_t$ is contained in $\sim^c$. In the presence of the equational characterization of $\sim_t$ given in Theorem 3.2, it is possible to give an elegant, algebraic proof of this fact. Let us recall that, by Theorem 3.2, $\sim_t$ is the least congruence over $\mathbf{P}$ which satisfies the set of equations in Fig. 1. Moreover, by the construction of $\sim^c$, $\sim^c$ is a congruence over $\mathbf{P}$. Hence, in order to show that $\sim_t \subseteq \sim^c$, it is sufficient to prove that $\sim^c$ satisfies all the equations which completely characterize $\sim_t$ over the language $\mathbf{P}$.

For the sake of completeness, let us recall that $\sim$ is a congruence with respect to all the combinators apart the action refinement one. Hence $\sim^c$ can simply be obtained by closing $\sim$ with respect to all the "refinement contexts." Formally, for each $p, q \in \mathbf{P}$,

$$p \sim^c q \text{ iff for all } \mathbf{P}_{\text{ref}}\text{-contexts } C[\cdot],\ C[p] \sim C[q]$$

$$\text{iff for all } a_0, \ldots, a_n \in \mathbf{A},\ r_0, \ldots, r_n \in \mathbf{P}_{\text{ref}},\ p\varrho \sim q\varrho,$$

where ϱ denoted the syntactic substitution $[a_0/\mathbf{red}(r_0)] \cdots [a_n/\mathbf{red}(r_n)]$. We may now give a first, algebraic proof of the fact that $\sim_t$ coincides with $\sim^c$ over $\mathbf{P}$.

THEOREM 3.4 (The Characterization Theorem: Algebraic Proof). *For each* $p, q \in \mathbf{P}$, $p \sim^c q$ *iff* $p \sim_t q$.

Proof. The "only if" implication follows by Corollary 3.1. In view of Theorem 3.2, to prove the "if" implication it is sufficient to show that, for each $p, q, r_0, \ldots, r_n \in \mathbf{P}$ and $a_0, \ldots, a_n \in \mathbf{A}$, whenever $p = q$ is an instance of an equation in Fig. 1 then

$$p[a_0/r_0] \cdots [a_n/r_n] \sim q[a_0/r_0] \cdots [a_n/r_n].$$

This follows easily for each equation in Fig. 1 because syntactic substitution is a homomorphism with respect to all the operators in Σ and by the soundness of all the equations with respect to $\sim$. For instance, let

$$(p \lfloor q) \lfloor t = p \lfloor (q \mid t)$$

be an instance of axiom (B2) and ϱ denote the syntactic substitution $[a_0/r_0]\cdots[a_n/r_n]$. Then

$$\begin{aligned}((p \mathbin{⅂} q) \mathbin{⅂} t)\varrho &= (p\varrho \mathbin{⅂} q\varrho) \mathbin{⅂} t\varrho \\ &\sim p\varrho \mathbin{⅂} (q\varrho \mid t\varrho) \qquad \text{as (B2) is sound with respect to } \sim \\ &= (p \mathbin{⅂} (q \mid t))\varrho.\end{aligned}$$

Thus, for each $p, q \in \mathbf{P}$, $p \sim^c q$ iff $p \sim_t q$. ∎

However, the above-given algebraic proof of the characterization theorem for $\sim^c$ is language-dependent and relies on the equational characterization of $\sim_t$ given in Theorem 3.2. Moreover, for the proof to work, it is essential that only processes and not actions be referred to in the set of complete equations. As it is unlikely that such a characterization will be available over more complex algebras, for instance those including a restriction operator [Mil80], it would be useful to give an alternative, *behavioural* proof of the fact that $\sim_t$ is contained in $\sim^c$ over $\mathbf{P}$. Such a proof will allow us both to shed more light on the properties of $\sim_t$ which make sure that such an equivalence is preserved by action-refinement and to introduce a wealth of operational techniques and results which have proved to be of considerable use in extending the work presented in this paper to richer algebras [AH90]. In order to give a behavioural proof of our claim that $\sim_t$ is contained in $\sim^c$ over $\mathbf{P}$, the main problem is to show that $\sim_t$ is preserved by the operation of action-refinement. We already know (Proposition 3.1) that $\sim_t$ is a congruence with respect to the other combinators of our calculus and that it is contained in $\sim$, so if we show that $\sim_t$ is preserved by $[a \rightsquigarrow q]$ the claim would follow by the fact that $\sim^c$ is the *largest congruence* contained in $\sim$. The proof of this result is much more involved and will be facilitated by the use of a version of timed equivalence, denoted by $\sim_r$ and called *refine-equivalence*, which is induced on $\mathscr{S}$ by a family of relations on a labelled version of the calculus.

4. Refine-Equivalence

4.1. *Motivation*

In proving that $\sim_t$ is a congruence with respect to $[a \rightsquigarrow q]$, the combinator for refining actions by processes, it will be technically useful to consider a version of our language in which we may safely talk about different occurrences of the same action symbol in a term.

The need for this technical complication can be explained by means of an example.

EXAMPLE 4.1. Assume we have specified a system by means of the process term $p = (a \mid a) \mid a$ and, at a lower level of abstraction, we want to refine a to $q = b : c$. As parallel processes may evolve asynchronously, $p[a \rightsquigarrow q]$ may indeed reach a stage of its evolution in which the three a-processes will each be at a (possibly) different stage of their evolution. Hence it is necessary to devise a method which allows us to express the fact that different "copies" of process q are currently active and to keep track of the state each of them has reached.

The solution we propose in this section is to consider a labelled version of our simple language. We restrict ourselves to considering labelled process terms in which each labelled action occurs at most once. The labelled calculus is endowed with a relation which captures a notion of bisimulation between labelled terms. Two labelled terms will be *bisimilar* if there is a kind of label-preserving correspondence between their labelled actions which is consistent with their dynamic behaviour. This notion of bisimulation on the labelled calculus induces an equivalence relation on the unlabelled one which we denote $\sim_r$ and call *refine-equivalence*. This new equivalence is of independent interest as it is an alternative formulation of the idea of strong bisimulation in the presence of non-instantaneous actions. We eventually show that $\sim_r$ and $\sim_t$ coincide, but we can show fairly straightforwardly that $\sim_r$ is preserved by action-refinement.

4.2. *The Calculus and Its Operational Semantics*

We now present the formal definition of the labelled calculus we use in proving that $\sim_t$ is a congruence with respect to action-refinement.

DEFINITION 4.1 (Uniquely Labelled Processes). (i) The set of *labelled atomic actions*, **LA**, is defined as follows:

$$\mathbf{LA} \underset{\text{def}}{=} \{a_i \mid a \in \mathbf{A} \wedge i \in \mathbf{N}\}.$$

(ii) The set of *uniquely labelled processes*, **LP**, is the set of terms generated by the grammar

$$\pi ::= nil \mid a_i \mid \pi + \pi \mid \pi ; \pi \mid \pi \mid \pi \mid \pi \setminus \pi,$$

where $a_i \in \mathbf{LA}$, subject to the constraint that each index i occurs at most once in π. **LP** is ranged over by $\pi, \pi', \pi_1, \ldots$.

DEFINITION 4.2 (Configurations). The set of *uniquely labelled states*, or *configurations*, $\mathbf{L}\mathcal{S}$, is defined by simply adapting the clauses of Definition 3.1 to the set of uniquely labelled processes, subject to the constraint that, for each configuration c, each index i occurs at most once in c. $\mathbf{L}\mathcal{S}$ will be ranged over by $c, d, c', d', \ldots$.

The operational semantics for labelled states is defined in terms of next-event relations $\stackrel{\lambda}{\longmapsto}$, where λ ranges over the set of *labelled subactions*

$$\mathbf{LA}_s \underset{\text{def}}{=} \{S(a_i) \mid a_i \in \mathbf{LA}\} \cup \{F(a_i) \mid a_i \in \mathbf{LA}\}.$$

The interpretation we impose on labelled subactions is a simple variant of the one we have imposed on the subactions in $\mathbf{A}_s$. For each $a_i \in \mathbf{LA}$, the i th occurrence of the action symbol a, $S(a_i)$ stands for the beginning of its execution and $F(a_i)$ stands for its termination.

The termination predicate on $\mathbf{L}\mathscr{S}$, needed in the definition of the operational semantics, and the operational semantics for $\mathbf{L}\mathscr{S}$ itself can be defined using the standard method. We leave the details to the reader who should also check the following result:

PROPOSITION 4.1. $\forall c \in \mathbf{L}\mathscr{S}, \lambda \in \mathbf{LA}_s\ c \stackrel{\lambda}{\longmapsto} c' \Rightarrow c' \in \mathbf{L}\mathscr{S}$.

Having developed an operational view of configurations we would like to say when two configurations c and d are *behaviourally equivalent*. The notion of equivalent behaviour between configurations is not absolute but is parameterized with respect to a correspondence between their component states of the form $F(a_i)$ for some $a_i \in \mathbf{LA}$. The technical definition takes the form of a parameterized timed bisimulation. The next few definitions introduce the technical machinery we will need in introducing this kind of relation.

The parameter in the definition of bisimulation for configurations will be a family of binary relations, one for each $a \in \mathbf{A}$, on $\mathbf{N}$. If ϕ is one of those families of relations and $c \sim_\phi d, c, d \in \mathbf{L}\mathscr{S}$, then, for each $(i, j) \in \phi_a$, $F(a_i)$ and $F(a_j)$ *play the same role* in the dynamic behaviour of the configurations.

The informal notion of *playing the same role* stated above will be formalized by the definition of bisimulation.

DEFINITION 4.3. (i) Let $\mathscr{R}$ be a binary relation over $\mathbf{N}$. Then $\mathscr{R}$ is (the graph of) a partial bijection on $\mathbf{N}$ iff it is a bijection between two subsets of $\mathbf{N}$.

(ii) $\mathscr{H}$ will denote the set of $\mathbf{A}$-indexed families of partial bijections over $\mathbf{N}$ and will be ranged over by $\phi, \varphi, \ldots$.

Notation 4.1. In what follows $\varnothing$ denotes both the empty set and, with abuse of notation, the $\mathbf{A}$-indexed family of relations such that

$$\forall a \in \mathbf{A}\ a \mapsto \varnothing.$$

We also sometimes write $(i, j) \in \phi_a$ as $(a, i, j) \in \phi$, where $\phi \in \mathscr{H}$. All the usual operations and predicates on relations are extended pointwise to indexed families of relations in the natural way.

For example, if $\mathscr{R}$ and $\mathscr{R}'$ are $\mathscr{H}$-indexed families of binary relations

$$\mathscr{R} \subseteq \mathscr{R}' \Leftrightarrow \forall \phi \in \mathscr{H}\ \mathscr{R}_\phi \subseteq \mathscr{R}'_\phi.$$

An A-indexed family of partial bijections $\phi \in \mathscr{H}$ is used to record information about which actions have been started and to whom they have been matched in the bisimulation functional. We call such information a *history*. Intuitively, the presence of (a, i, j) in a history ϕ means that in one process action a_i has started, in the other action a_j has started and we are expecting the ith occurrence of a in one to be matched to the jth in the other. The matching between actions a_i and a_j is recorded in a history h only until the two actions terminate. From that moment onwards the information about this matching becomes redundant and is discarded from h. Thus histories will only record information about the matching between the actions which are currently being executed rather than the whole set of such associations.

Given a configuration c, any $F(a_i)$ occurring in c is called a *place* of c. Intuitively $F(a_i)$ stands for a state in which the execution of the atomic labelled process a_i has started but has not terminated yet. The set of a-places in c is defined as follows:

DEFINITION 4.4. For each $a \in \mathbf{A}$, $c \in \mathbf{L}\mathscr{S}$

$$\mathbf{Places}(a, c) \underset{\text{def}}{=} \{i \mid F(a_i) \text{ occurs in } c\}.$$

Configurations are equivalent or inequivalent with respect to a history ϕ. For c and d to be equivalent with respect to $\phi \in \mathscr{H}$, we require that

- all actions which have started in c or d and are not yet finished must be recorded in ϕ;
- every start move of c, $c \xmapsto{S(a_i)} c'$, must be matched by a corresponding move $d \xmapsto{S(a_j)} d'$ of d such that c' and d' are equivalent with respect to the augmented history $\phi \cup \{(a, i, j)\}$; and
- every end move from c, $c \xmapsto{F(a_i)} c'$, must be matched by the end move of d associated with it in the history ϕ, i.e., $d \xmapsto{F(a_j)} d'$ with $(a, i, j) \in \phi$ and c' and d' are equivalent with respect to the diminished history $\phi - \{(a, i, j)\}$.

This is the motivation underlying the following formal definition.

DEFINITION 4.5 (Parameterized Bisimulation). 1. Let $\mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$ denote the set of $\mathscr{H}$-indexed families of binary relations over $\mathbf{L}\mathscr{S}$.

2. Define a functional $\mathscr{G}: \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}} \to \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$ as follows: given $\mathscr{R} \in \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$, $(c_1, c_2) \in \mathscr{G}(\mathscr{R})_\phi$ iff

(i) $\forall a \in \mathbf{A}\ \mathrm{dom}(\phi_a) = \mathbf{Places}(a, c_1)$ and $\mathrm{range}(\phi_a) = \mathbf{Places}(a, c_2)$

(ii) $\forall a_i \in \mathbf{LA}$

(a) $c_1 \overset{S(a_i)}{\longmapsto} c_1' \Rightarrow \exists j \in \mathbf{N},\ c_2' \in \mathbf{L}\mathscr{S}: c_2 \overset{S(a_j)}{\longmapsto} c_2'$ and $(c_1', c_2') \in \mathscr{R}_{\phi \cup \{(a,i,j)\}}$

(b) $c_1 \overset{F(a_i)}{\longmapsto} c_1' \Rightarrow \exists c_2' \in \mathbf{L}\mathscr{S}: \phi_a(i) = j$ and $c_2 \overset{F(a_j)}{\longmapsto} c_2'$ and $(c_1', c_2') \in \mathscr{R}_{\phi - \{(a,i,j)\}}$

(iii) $\forall a_j \in \mathbf{LA}$

(a) $c_2 \overset{S(a_j)}{\longmapsto} c_2' \Rightarrow \exists i \in \mathbf{N},\ c_1' \in \mathbf{L}\mathscr{S}: c_1 \overset{S(a_i)}{\longmapsto} c_1'$ and $(c_1', c_2') \in \mathscr{R}_{\phi \cup \{(a,i,j)\}}$

(b) $c_2 \overset{F(a_j)}{\longmapsto} c_2' \Rightarrow \exists c_1': \phi_a(i) = j$ and $c_1 \overset{F(a_i)}{\longmapsto} c_1'$ and $(c_1', c_2') \in \mathscr{R}_{\phi - \{(a,i,j)\}}$

3. $\mathscr{R} \in \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$ is a parameterized bisimulation iff $\mathscr{R} \subseteq \mathscr{G}(\mathscr{R})$.

$$\forall \phi \in \mathscr{H} \sim_\phi \underset{\text{def}}{=} \bigcup \{\mathscr{R}_\phi \mid \mathscr{R} \subseteq \mathscr{G}(\mathscr{R})\}.$$

$(\mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}, \subseteq)$ is a complete lattice as $\subseteq$ is induced pointwise by the relation of set inclusion. The following result is then standard.

PROPOSITION 4.2. (i) *$\mathscr{G}(\cdot)$ is a monotonic endofunction over the complete lattice $(\mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}, \subseteq)$.*

(ii) *$\{\sim_\phi \mid \phi \in \mathscr{H}\}$ is the maximum fixed-point of $\mathscr{G}(\cdot)$.*

4.3. *Definition of Refine-Equivalence*

The maximum fixed-point of the functional $\mathscr{G}(\cdot)$, $\{\sim_\phi \mid \phi \in \mathscr{H}\}$, can be used to induce an equivalence relation on the set $\mathscr{S}$ of unlabelled states. This relation is the *refine-equivalence* promised in the previous section and is a technical tool used to give a behavioural proof of the fact that $\sim_t$ is preserved by $[a \rightsquigarrow q]$. The definition is very similar to that of timed-equivalence. There is only the additional constraint that occurrences of actions should be properly matched; namely, whenever the start of the i th occurrence of an action is matched by the start of the j th occurrence then the finish of the j th occurrence must also be used to match the finish of the i th occurrence.

This definition is more in accordance with the intuitive idea underlying strong observational equivalence in the presence of non-instantaneous actions. We still want to match the ability of processes to perform actions and their resulting potential behaviour and, to ensure that complete actions are compared, we should consider not only their beginnings and endings, but also their entire duration. This is achieved by using histories.

DEFINITION 4.6. 1. The function $\mathbf{un}(\cdot): \mathbf{L}\mathscr{S} \mapsto \mathscr{S}$ is defined as the unique homomorphism between the two algebras such that

- $\mathbf{un}(nil) = nil$
- $\mathbf{un}(a_i) = a$
- $\mathbf{un}(F(a_i)) = F(a)$.

2. $\forall s_1, s_2 \in \mathscr{S}\ s_1 \sim_r s_2 \Leftrightarrow \exists c_1, c_2 \in \mathbf{L}\mathscr{S}: \mathbf{un}(c_i) = s_i, i = 1, 2$, and $\exists \phi \in \mathscr{H}$: $c_1 \sim_\phi c_2$.

Note. In what follows we often make use of the obvious fact that $\mathbf{un}(\cdot)$ is onto.

Notation 4.2. From now on, if $e \in \mathbf{A}_s$ and $i \in \mathbf{N}$ then e_i stands for $S(a_i)$, if $e = S(a)$, $F(a_i)$, if $e = F(a)$.

The following lemma relates the derivations of a configuration c to those of its corresponding unlabelled state $\mathbf{un}(c)$.

LEMMA 4.1. *Let* $c \in \mathbf{L}\mathscr{S}$. *Then*

(i) $c\surd \Leftrightarrow \mathbf{un}(c)\surd$.

(ii) $\forall e_i \in \mathbf{LA}_s c \stackrel{e_i}{\longmapsto} c' \Rightarrow \mathbf{un}(c) \stackrel{e}{\Longrightarrow} \mathbf{un}(c')$.

(iii) $\forall e \in \mathbf{A}_s \mathbf{un}(c) \stackrel{e}{\Longrightarrow} s \Rightarrow \exists c', j: c \stackrel{e_j}{\longmapsto} c' \wedge \mathbf{un}(c') = s$.

Proof. By structural induction on c. Omitted. ∎

One simple corollary of this lemma is the fact that $\sim_r$ is contained in $\sim_t$.

THEOREM 4.1. $(\sim_r \subseteq \sim_t): \forall s_1, s_2 \in \mathscr{S}\ s_1 \sim_r s_2 \Rightarrow s_1 \sim_t s_2$.

Proof. Define the relation $\mathscr{R} \subseteq \mathscr{S}^2$:

$$\mathscr{R} \underset{\text{def}}{=} \{(\mathbf{un}(c), \mathbf{un}(d)) \mid \exists \phi \in \mathscr{H}: c \sim_\phi d\}.$$

We show that $\mathscr{R}$ is a timed-bisimulation. By symmetry, it is sufficient to show that, for each $(\mathbf{un}(c), \mathbf{un}(d)) \in \mathscr{R}$, $\mathbf{un}(c) \stackrel{e}{\Longrightarrow} s \Rightarrow \exists s'$: $\mathbf{un}(d) \stackrel{e}{\Longrightarrow} s' \wedge (s, s') \in \mathscr{R}$.

Assume that $\mathbf{un}(c) \stackrel{S(a)}{\Longrightarrow} s$. Then, by Lemma 4.1,

$$\exists i, c': c \stackrel{S(a_i)}{\longmapsto} c' \wedge \mathbf{un}(c') = s.$$

As $c \sim_\phi d$ for some $\phi \in \mathscr{H}$,

$$\exists j, d': d \stackrel{S(a_j)}{\longmapsto} d' \wedge c' \sim_{\phi \cup \{(a,i,j)\}} d'.$$

By Lemma 4.1, $d \overset{S(a_j)}{\longmapsto} d'$ implies $\mathbf{un}(d) \overset{S(a)}{\Longrightarrow} \mathbf{un}(d')$. Moreover, by the definition of $\mathscr{R}$, $(\mathbf{un}(c'), \mathbf{un}(d')) \in \mathscr{R}$.

Similarly one can match the $F(a)$-moves of the configuration c.

Hence $\mathscr{R}$ is a timed-bisimulation.

Let $s_1, s_2 \in \mathscr{S}$ be such that $s_1 \sim_r s_2$. Then, by the definition of $\sim_r$, $s_1 \sim_r s_2$ implies $(s_1, s_2) \in \mathscr{R}$. It follows that $s_1 \sim_t s_2$. ∎

We now show that $\sim_r$ is an equivalence relation on $\mathscr{S}$.

THEOREM 4.2. *$\sim_r$ is an equivalence relation on $\mathscr{S}$.*

Proof. We show that $\sim_r$ is reflexive and transitive leaving the proof that it is symmetric to the reader.

Reflexivity. Let $1_I =_{\text{def}} \{(i, i) \mid i \in I\}$, for $I \subseteq \mathbf{N}$. A family of relations ϕ in $\mathscr{H}$ is an identity if and only if, for each $a \in \mathbf{A}$, $\phi_a = 1_I$ for some $I \subseteq \mathbf{N}$.

Define $\mathscr{R} \in \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$ as follows:

$$\mathscr{R}_\phi \underset{\text{def}}{=} \begin{cases} \varnothing & \text{if } \phi \text{ is not an identity} \\ \{(c, c) \mid \forall a \in \mathbf{A}\ \mathbf{Places}(a, c) = \mathrm{dom}(\phi_a)\} & \text{otherwise.} \end{cases}$$

It can be easily seen that $\mathscr{R}$ is a parameterized bisimulation.

As $\forall s \in \mathscr{S}\ \exists c \in \mathbf{L}\mathscr{S}: \mathbf{un}(c) = s$ and $(c, c) \in \mathscr{R}_\phi$, where ϕ is such that

$$\forall a \in \mathbf{A} \phi_a = \{(i, i) \mid i \in \mathbf{Places}(a, c)\},$$

we have that $c \sim_\phi c$ and $s \sim_r s$.

Transitivity. Define the following $\mathscr{R} \in \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$:

$$\forall \phi \in \mathscr{H}\ \mathscr{R}_\phi \underset{\text{def}}{=} \bigcup \{ \sim_\psi \circ \sim_\varphi \mid \phi = \psi \circ \varphi \wedge \psi, \varphi \in \mathscr{H} \}.$$

It can be shown that $\mathscr{R}$ is a parameterized bisimulation.

Moreover, $s_1 \sim_r s_2$ and $s_2 \sim_r s_3$ imply

$$\exists c_i, i = 1, 2, 3, : \mathbf{un}(c_i) = s_i \text{ and } \exists \psi, \varphi: c_1 \sim_\varphi c_2 \wedge c_2 \sim_\psi c_3.$$

Hence $(c_1, c_3) \in \mathscr{R}_{\psi \circ \varphi}$. This implies $c_1 \sim_{\psi \circ \varphi} c_3$ and $s_1 \sim_r s_3$. ∎

Note that from the definition of $\sim_\phi$ it follows easily that

$$\forall \pi_1, \pi_2 \in \mathbf{LP}\ \pi_1 \sim_\phi \pi_2 \Rightarrow \phi = \varnothing.$$

Hence, for all $p, q \in \mathbf{P}$, $p \sim_r q$ if, and only if,

$$\exists \pi_1, \pi_2: \mathbf{un}(\pi_1) = p \wedge \mathbf{un}(\pi_2) = q \wedge \pi_1 \sim_\varnothing \pi_2.$$

We now show that in the definition of $\sim_r$ we could have used a *for all* instead of $\exists c_1, c_2, \ldots$. We hope that this result adds to the intuition of the definition of parameterized bisimulation. To do so we first show that if c and d are configurations which reduce to the same unlabelled state $s \in \mathscr{S}$ then there exists a $\phi \in \mathscr{H}$ such that $c \sim_\phi d$. First of all we prove a technical lemma which is needed in what follows.

LEMMA 4.2. *Let $c, d \in \mathbf{L}\mathscr{S}$. Assume $\mathbf{un}(c) = \mathbf{un}(d)$ and $c \xmapsto{F(a_i)} c'$. Then there exists a unique $j \in \mathbf{N}$, $d' \in \mathbf{L}\mathscr{S}$ such that*

$$d \xmapsto{F(a_j)} d' \qquad \textit{and} \qquad \mathbf{un}(c') = \mathbf{un}(d').$$

Proof. By induction on the structure of c.

$c = \pi$. Vacuous.

$c = F(a_i)$. Then d is of the form $F(a_j)$ for some $j \in \mathbf{N}$. The result follows trivially.

$c = c_1; \pi$. Then d is of the form $d_1; \pi_1$ with $\mathbf{un}(c_1) = \mathbf{un}(d_1)$ and $\mathbf{un}(\pi) = \mathbf{un}(\pi_1)$. Assume $c_1; \pi \xmapsto{F(a_i)} c_1'; \pi$ because $c_1 \xmapsto{F(a_i)} c_1'$.

By the inductive hypothesis, $\exists! j \in \mathbf{N}, \exists! d_1' : d_1 \xmapsto{F(a_j)} d_1'$ and $\mathbf{un}(c_1') = \mathbf{un}(d_1')$. Hence $d_1'; \pi_1$ and j are unique such that

$$d_1; \pi_1 \xmapsto{F(a_j)} d_1'; \pi_1 \qquad \text{and} \qquad \mathbf{un}(c_1'; \pi) = \mathbf{un}(d_1'; \pi_1).$$

$c = c_1 \mid c_2$. Similar. ∎

Lemmas 4.1–4.2 will also be useful in deriving a parameterized bisimulation $\mathscr{R}(c, d)$ and $\phi(c, d) \in \mathscr{H}$ such that $(c, d) \in \mathscr{R}(c, d)_{\phi(c,d)}$. The next definition, which is rather technical, is devoted to the construction of such a parameterized bisimulation. The construction is by induction on the size of c (and therefore also of d as $\mathbf{un}(c) = \mathbf{un}(d)$).

DEFINITION 4.7. Let $c, d \in \mathbf{L}\mathscr{S}$. Assume $\mathbf{un}(c) = \mathbf{un}(d)$. The following inductive construction is given:

- If $c\surd$ and $d\surd$ then $\mathscr{R}(c, d) \in \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$ is defined as follows:

$$\forall \phi \in \mathscr{H}\ \mathscr{R}(c, d)_\phi \underset{\text{def}}{=} \begin{cases} \{(c, d)\} & \text{if } \phi = \varnothing \\ \varnothing & \text{otherwise.} \end{cases}$$

Moreover, $\phi(c, d) = \varnothing$.

- If c and d are not terminated then define the following sets:

$$\mathbf{S} \underset{\text{def}}{=} \{(c', d') \mid \exists S(a_i), S(a_j) : c \xmapsto{S(a_i)} c' \wedge d \xmapsto{S(a_j)} d' \wedge \mathbf{un}(c') = \mathbf{un}(d')\}$$

$$\mathbf{F} \underset{\text{def}}{=} \{(c', d') \mid \exists F(a_i), F(a_j) : c \xmapsto{F(a_i)} c' \wedge d \xmapsto{F(a_j)} d' \wedge \mathbf{un}(c') = \mathbf{un}(d')\}$$

$$\mathbf{X} \underset{\text{def}}{=} \{(F(a_i), F(a_j)) \mid c \xmapsto{F(a_i)} c' \wedge d \xmapsto{F(a_j)} d' \wedge \mathbf{un}(c') = \mathbf{un}(d')\}.$$

Note that, by Lemma 4.2, **F** and **X** are graphs of injective functions.

Then $\phi(c, d) \underset{\text{def}}{=} \{(a, i, j) \mid (F(a_i), F(a_j)) \in \mathbf{X}\}$ and $\mathscr{R}(c, d)$ is defined as follows:

$$\forall \phi \in \mathscr{H}\ \mathscr{R}(c, d)_\phi \underset{\text{def}}{=} \begin{cases} \bigcup_{(c', d') \in \mathbf{S} \cup \mathbf{F}} \mathscr{R}(c', d')_\phi & \text{if } \phi \neq \phi(c, d) \\ \bigcup_{(c', d') \in \mathbf{S} \cup \mathbf{F}} \mathscr{R}(c', d')_\phi \cup \{(c, d)\} & \text{if } \phi = \phi(c, d). \end{cases}$$

Note that in the above given construction the sets **S**, **F**, **X** are always finite. Moreover, the size of c' (and therefore also of d') has decreased. Before showing that the construction is indeed correct we state the following lemma.

LEMMA 4.3. *For each* $c, d \in \mathbf{L}\mathscr{S}$ *such that* $\mathbf{un}(c) = \mathbf{un}(d)$ *the following statements hold:*

1. $c \xmapsto{S(a_i)} c'$ *and* $d \xmapsto{S(a_j)} d'$ *and* $\mathbf{un}(c') = \mathbf{un}(d')$ *implies* $\phi(c', d') = \phi(c, d) \cup \{(a, i, j)\}$.
2. $c \xmapsto{F(a_i)} c'$ *and* $d \xmapsto{F(a_j)} d'$ *and* $\mathbf{un}(c') = \mathbf{un}(d')$ *implies* $\phi(c', d') = \phi(c, d) - \{(a, i, j)\}$.

Proof. By structural induction on $\mathbf{un}(c)$. Omitted. ∎

PROPOSITION 4.3. *For each* $c, d \in \mathbf{L}\mathscr{S}$ *such that* $\mathbf{un}(c) = \mathbf{un}(d)$, $\mathscr{R}(c, d)$ *is a parameterized bisimulation.*

Proof. By induction on the size of c (and therefore also of d).

The base case, $c\surd$ and $d\surd$, is obvious.

Assume c is not terminated. By inductive hypothesis, we know that for each $(c', d') \in \mathbf{S} \cup \mathbf{F}$, $\mathscr{R}(c', d')$ is a parameterized bisimulation.

It is therefore sufficient to show that $(c, d) \in \mathscr{G}(\mathscr{R}(c, d))_{\phi(c,d)}$.

Assume $c \xmapsto{S(a_i)} c'$. Then, by Lemma 4.1, $\mathbf{un}(c) \xRightarrow{S(a)} \mathbf{un}(c')$ and this implies the existence of j, d' such that $d \xmapsto{S(a_j)} d'$ and $\mathbf{un}(d') = \mathbf{un}(c')$.

By induction, $(c', d') \in \mathscr{R}_{\phi(c', d')}$.

Also, by the above lemma, $\phi(c', d') = \phi(c, d) \cup \{(a, i, j)\}$ and therefore $\mathbf{Places}(a, c) = \mathrm{dom}(\phi_a(c, d))$ and $\mathbf{Places}(a, d) = \mathrm{range}(\phi_a(c, d))$, since the corresponding property holds of $\phi(c', d')$.

The argument for the case when $c \xmapsto{F(a_i)} c'$ is similar.

Hence, by symmetry, $\mathscr{R}(c, d)$ is a parameterized bisimulation. ∎

As a corollary we get the following result which shows that in the definition of $\sim_r$ we could have made do with a *for all* instead of *there exist*.

COROLLARY 4.1. *The followig statements hold*:

(i) $\forall c, d \in \mathbf{L}\mathscr{S}$ $\mathbf{un}(c) = \mathbf{un}(d) \Rightarrow c \sim_{\phi(c,d)} d$.

(ii) *Let* $s_1, s_2 \in \mathscr{S}$. *Then* $s_1 \sim_r s_2$ *iff for all* $c_1, c_2 \in \mathbf{L}\mathscr{S}$ *such that* $\mathbf{un}(c_1) = s_1$ *and* $\mathbf{un}(c_2) = s_2$, *there exists* $\phi \in \mathscr{H}$ *such that* $c_1 \sim_\phi c_2$.

4.4. *Refine-Equivalence and Timed-Equivalence Coincide*

This section is entirely devoted to showing that, for our language, $\sim_r$ and $\sim_t$ coincide. However, contrary to what happens with $\sim_t$, it is quite straightforward to give a behavioural proof of the fact that $\sim_r$ is preserved by action-refinement. We have already shown that $\sim_r$ is contained in $\sim_t$. Unfortunately the converse is rather involved. Its proof is developed in two steps. First we show that $\sim_r$ is a congruence with respect to $+, ;, |, \backslash$. Then we show that $\sim_r$ satisfies all the axioms which completely characterize $\sim_t$ over our language (the axioms are collected in Fig. 1). The completeness theorem (Theorem 3.2) tells us that $\sim_t$ is the *least congruence* on $\mathbf{P}$ which satisfies the axioms; hence we will have shown the desired inclusion.

THEOREM 4.3. $\sim_r$ *is a* $\mathbf{P}$*-congruence.*

Proof. We will show that $\sim_t$ is preserved by $|$ leaving the other cases to the reader. Assume $s_1 \sim_r s_2$. Then, by definition of $\sim_r$,

$$\exists c_1, c_2 \in \mathbf{L}\mathscr{S} : \mathbf{un}(c_i) = s_i,\ i = 1, 2, \text{ and } \exists \phi \in \mathscr{H} : c_1 \sim_\phi c_2.$$

Let $s \in \mathscr{S}$. Then there exists $c \in \mathbf{L}\mathscr{S}$ such that $(c_1 \mid c, c_2 \mid c) \in \mathbf{L}\mathscr{S}^2$ and $\mathbf{un}(c) = s$. Define $\mathscr{R} \in \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$ as follows:

$$\forall \phi \in \mathscr{H}\ \mathscr{R}_\phi \underset{\text{def}}{=} \{(c \mid c', d \mid d') \in \mathbf{L}\mathscr{S}^2 \mid \exists \phi_1, \phi_2 \in \mathscr{H} : \phi = \phi_1 \cup \phi_2 \wedge c \sim_{\phi_1} d \wedge c' \sim_{\phi_2} d'\}.$$

We show that $\mathscr{R}$ is a parameterized bisimulation. Assume that $(c \mid c', d \mid d') \in \mathscr{R}_\phi$. Then there exist $\phi_1, \phi_2 \in \mathscr{H} : \phi_1 \cup \phi_2 = \phi$ and $c \sim_{\phi_1} d$ and $c' \sim_{\phi_2} d'$. Note that, as $(c \mid c', d \mid d') \in \mathbf{L}\mathscr{S}^2$, $\phi_1 \cap \phi_2 = \varnothing$.

(i) We show that clause (i) of the definition of parameterized bisimulation holds. This is because

$$\forall a \in \mathbf{A}\ \mathbf{Places}(a, c \mid c') = \mathbf{Places}(a, c) \cup \mathbf{Places}(a, c') = \mathrm{dom}(\phi_{1_a}) \cup \mathrm{dom}(\phi_{2_a}).$$

Moreover $\mathrm{dom}(\phi_a) = \mathrm{dom}(\phi_{1_a}) \cup \mathrm{dom}(\phi_{2_a})$. Similarly for the range of the relation ϕ_a.

(ii) Assume, without loss of generality, that $c \mid c' \xmapsto{S(a_i)} \bar{c} \mid c'$ because $c \xmapsto{S(a_i)} \bar{c}$. Then $\exists j, \bar{d} : d \xmapsto{S(a_j)} \bar{d}$ and $(\bar{c}, \bar{d}) \in \phi_1 \cup \{(a, i, j)\}$. This implies

$d \mid d' \xmapsto{S(a_j)} \bar{d} \mid d'$. Moreover, by the property of the operational semantics, $(\bar{c} \mid c', \bar{d} \mid d') \in \mathbf{L}\mathscr{S}^2$ and $(\bar{c} \mid c', \bar{d} \mid d') \in \mathscr{R}_{\phi_1 \cup \phi_2 \cup \{(a,i,j)\}}$, by definition of $\mathscr{R}$.

(iii) Assume, wlog, that $c \mid c' \xmapsto{F(a_i)} \bar{c} \mid c'$ because $c \xmapsto{F(a_i)} \bar{c}$.

As $c \sim_{\phi_1} d$ by hypothesis, $\exists \bar{d}: d \xmapsto{S(a_j)} \bar{d}$ and $\phi_{1_a}(i) = j$ and $(\bar{c}, \bar{d}) \in \sim_{\phi_1 - \{(a,i,j)\}}$. This implies $d \mid d' \xmapsto{F(a_j)} \bar{d} \mid d'$. Moreover, as $\phi_{1_a} \cap \phi_{2_a} = \varnothing$, $(\phi_1 \cup \phi_2)_a(i) = j$.

As $(\bar{c} \mid c', \bar{d} \mid d') \in \mathbf{L}\mathscr{S}^2$ we have that, by definition of $\mathscr{R}$, $(\bar{c} \mid c', \bar{d} \mid d') \in \mathscr{R}_{(\phi_1 - \{(a,i,j)\}) \cup \phi_2}$.

Moreover, as $\phi_1 \cap \phi_2 = \varnothing$,

$$(\phi_1 - \{(a, i, j)\}) \cup \phi_2 = (\phi_1 \cup \phi_2) - \{(a, i, j)\}.$$

By symmetry $\mathscr{R}$ is a parameterized bisimulation.

As, by definition of $\mathscr{R}$, $(c_1 \mid c, c_2 \mid c) \in \mathscr{R}_\psi$, where

$$\forall a \in \mathbf{A}\ \psi_a \underset{\text{def}}{=} \phi_a \cup \{(i, i) \mid i \in \mathbf{Places}(a, c)\},$$

we conclude that $c_1 \mid c \sim_\psi c_2 \mid c$. Hence $s_1 \mid s \sim_r s_2 \mid s$. ∎

To complete the proof of the inclusion $\sim_t \subseteq \sim_r$, we need to show that $\sim_r$ satisfies all the laws of the complete axiomatization of $\sim_t$ on the language **P**. The equational characterization of $\sim_t$ is given in Fig. 1.

THEOREM 4.4. *Let $\mathscr{A}$ denote the set of axioms in Fig. 1. Then $t = t' \in \mathscr{A}$ implies that $t = t'$ is valid for $\sim_r$.*

Proof. The proof of this result is rather technical and tedious. For instance, when proving the validity of axioms **B1** and **X1** care must be taken in making "copies" of x, y, and z with disjoint labellings.

The techniques developed in Section 4 are very useful in this respect. We examine only the axiom **B2**, which allows us to show the general proof technique. The remaining axioms are left to the reader.

B2. $(x \between y) \between z = x \between (y \mid z)$. Define $\mathscr{R} \in \mathscr{P}(\mathbf{L}\mathscr{S}^2)^{\mathscr{H}}$ as follows,

$$\forall \phi \in \mathscr{H}\ \mathscr{R}_\phi \underset{\text{def}}{=} \{((d_1 \between d_2) \between d_3, d_1 \between (d_2 \mid d_3)),$$
$$((d_1 \mid d_2) \mid d_3, d_1 \mid (d_2 \mid d_3)) \in \mathbf{L}\mathscr{S}^2 \mid \Gamma\},$$

where Γ is the predicate, a function of d_1, d_2, d_3, ϕ, defined as follows,

$$\forall a \in \mathbf{A}\ \phi_a = \left\{(i, i) \;\middle|\; i \in \bigcup_{j=1}^{3} \mathbf{Places}(a, d_j)\right\}.$$

We show that $\mathscr{R}$ is a parameterized bisimulation.

Assume $((d_1 \lfloor d_2) \lfloor d_3, d_1 \lfloor (d_2 \mid d_3)) \in \mathscr{R}_\phi$ and $(d_1 \lfloor d_2) \lfloor d_3 \xrightarrow{S(a_i)} (d_1' \mid d_2) \mid d_3$. Then $d_1 \xrightarrow{S(a_i)} d_1'$.

This implies $d_1 \lfloor (d_2 \mid d_3) \xrightarrow{S(a_i)} d_1' \mid (d_2 \mid d_3)$.

By the definition of $\mathscr{R}$, $((d_1' \mid d_2) \mid d_3, d_1' \mid (d_2 \mid d_3)) \in \mathscr{R}_{\phi \cup \{(a,i,i)\}}$.

Similarly for moves of the form $(d_1 \lfloor d_2) \lfloor d_3 \xrightarrow{F(a_i)} (d_1' \mid d_2) \mid d_3$.

The case $((d_1 \mid d_2) \mid d_3, d_1 \mid (d_2 \mid d_3))$ follows a similar pattern.

Hence $\mathscr{R}$ is a parameterized bisimulation and this is sufficient to show the validity of axiom **B2** for $\sim_r$. ▮

Finally we may state the main result of this section, namely that $\sim_r$ and $\sim_t$ coincide.

THEOREM 4.5. $\forall s_1, s_2 \in \mathscr{S}\ s_1 \sim_t s_2 \Leftrightarrow s_1 \sim_r s_2$.

Proof. Follows from Theorems 4.1, 4.3 and 4.4. ▮

4.5. *Refine-Equivalence Is Preserved by Action-Refinement*

This section is devoted to giving a behavioural proof of the fact that $\sim_r$ is preserved by action-refinement. The proof is articulated in several steps. First we define a suitable notion of substitution for labelled terms. Then we proceed by studying the relationships between the moves of a configuration c to which a substitution ρ has been applied and those of $\mathbf{un}(c)$ and ρ.

We conclude the section by showing that $\sim_r$, and thus $\sim_t$, is a congruence with respect to substitution.

DEFINITION 4.8. A substitution ρ is a mapping $\rho: \mathbf{LA} \cup \{F(a_i) \mid \mathbf{LA}\} \to \mathscr{S}$ such that:

(i) $\forall a_i \in \mathbf{LA}\ \rho(a_i) \in \mathbf{P}$;

(ii) $\forall i, j \in \mathbf{N}\ \rho(a_i) = \rho(a_j)$.

Let **SUB** denote the set of all the substitutions.

Note that we require actions to be substituted with processes and not with states (this satisfies our intuition that an action stands for a task which has not started evolving yet), but, for instance, we allow actions to be mapped to terminated processes. This fact has interesting implications which we hope to explore in more detail in the future.

EXAMPLE 4.2. Consider $p = (a_1; b_1) + (a_2; c_1)$. This process will choose non-deterministically one of the two branches of the computation it stands for when presented with the action a (remember that a_1 and a_2 are considered as two occurrences of action a). Moreover, this choice is

made internally by the process itself and cannot be influenced by the experimenter. This is an example of *internal non-determinism.*

Now let ρ be a substitution mapping a_1 and a_2 to nil and renaming b_1 to b and c_1 to c. Then $p\rho = (nil; b) + (nil; c)$. By means of the substitution we have hidden the a-actions and, in doing so, we have made the choice of the branch of the computation controllable by the environment. We have made the non-determinism exhibited by the process *external.*

This power of process substitution makes it more difficult to relate the moves of $c\rho$ to those of c and those of ρ. To do so we need a formal method for dealing with the situations described by the above example.

We define a way of associating to each configuration c a set of couples of the form (A, d), where A is a set of labelled actions and states of the form $F(a_i)$. The intuition captured by these couples is the following: given $\rho \in \mathbf{SUB}$, if (A, d) is associated to c and $\rho(A)\surd$ then $c\rho$ behaves like $d\rho$.

Note. For simplicity, in the following definition of the functions η and η' we will only give the cases which are needed in the formal developments presented in this section. In fact, as we mostly prove our technical results by some form of induction, it is sufficient to define the effect of a substitution ρ on the initial moves of a configuration.

DEFINITION 4.9. 1. The function η is defined by structural induction on π as follows:

(i) $\eta(nil) = \{(\varnothing, nil)\}$

(ii) $\eta(a_i) = \{(\{a_i\}, nil)\}$

(iii) $\eta(\pi_1 + \pi_2) = \{(x, \pi' + \pi_2) \mid (x, \pi') \in \eta(\pi_1)\} \cup \{(x, \pi_1 + \pi') \mid (x, \pi') \in \eta(\pi_2)\}$

(iv) $\eta(\pi_1; \pi_2) = \begin{cases} \eta(\pi_2) & \text{if } \pi_1\surd \\ \{(x, \pi'; \pi_2) \mid (x, \pi') \in \eta(\pi_1)\} & \text{otherwise} \end{cases}$

(v) $\eta(\pi_1 \mid \pi_2) = \{(x, \pi' \mid \pi_2) \mid (x, \pi') \in \eta(\pi_1)\} \cup \{(x, \pi_1 \mid \pi') \mid (x, \pi') \in \eta(\pi_2)\}$

(vi) $\eta(\pi_1 \between \pi_2) = \{(x, \pi' \between \pi_2) \mid (x, \pi') \in \eta(\pi_1)\}$,

2. The function η' is the extension of η to configurations. It is defined by structural induction on c as follows:

(i) $\eta'(\pi) = \eta(\pi)$

(ii) $\eta'(F(a_i)) = \{(\{F(a_i)\}, nil)\}$

(iii) $\eta'(c; \pi) = \begin{cases} \eta(\pi) & \text{if } c\surd \\ \{(x, c'; \pi) \mid (x, c') \in \eta'(c)\} & \text{otherwise} \end{cases}$

(iv) $\eta'(c_1 \mid c_2) = \{(x, c_1' \mid c_2) \mid (x, c_1') \in \eta'(c_1)\} \cup \{(x, c_1 \mid c_2') \mid (x, c_2') \in \eta'(c_2)\}$.

Notation 4.3. Let $X \subseteq \mathbf{LA} \cup \{F(a_i) \mid a_i \in \mathbf{LA}\}$. Then

$$\rho(X)\surd \Leftrightarrow \forall a_i, F(a_i) \in X\ \rho(a_i)\surd \wedge \rho(F(a_i))\surd.$$

The following proposition and corollary state formally that what we hoped for in giving the above definition actually holds.

In fact, if $(X, d) \in \eta'(c)$ and $\rho(X)\surd$ then $c\rho \sim_t d\rho$.

PROPOSITION 4.4. $\forall \pi \in \mathbf{LP}, \rho \in \mathbf{SUB},\ (X, \pi') \in \eta(\pi) \wedge \rho(X)\surd \Rightarrow \pi\rho \sim_t \pi'\rho$.

Proof. By structural induction on π. We give just the proof of the case $\pi = \pi_1 \mid \pi_2$ leaving the others to the reader.

$\pi = \pi_1 \mid \pi_2$. Assume $(X, \pi') \in \eta(\pi_1 \mid \pi_2)$. There are three cases to examine:

(a) $(X, \pi') = (X_1 \cup X_2, \pi_1' \mid \pi_2')$ with $(X_1, \pi_1') \in \eta(\pi_1)$ and $(X_2, \pi_2') \in \eta(\pi_2)$. By inductive hypothesis, $\mu_1\rho \sim_t \pi_1'\rho$ and $\pi_2\rho \sim_t \pi_2'\rho$. Since $\sim_t$ is a congruence it follows that $(\pi_1 \mid \pi_2)\rho = \pi_1\rho \mid \pi_2\rho \sim_t \pi_1'\rho \mid \pi_2'\rho$.

(b) $(X, \pi') = (X, \pi_1' \mid \pi_2)$ with $(X, \pi_1') \in \eta(\pi_1)$. By inductive hypothesis, $\pi_1\rho \sim_t \pi_1'\rho$. Again this means that $(\pi_1 \mid \pi_2)\rho = \pi_1\rho \mid \pi_2\rho \sim_t \pi_1'\rho \mid \pi_2\rho = (\pi_1' \mid \pi_2)\rho$.

(c) Symmetrical. ∎

COROLLARY 4.2. $\forall c \in \mathbf{L}\mathscr{S}, \rho \in \mathbf{SUB}\ (X, c') \in \eta'(c) \wedge \rho(X)\surd \Rightarrow c\rho \sim_t c'\rho$.

Proof. By structural induction on c. Omitted. ∎

With the next theorem we start the investigation of the possible origin of a move of $c\rho$, for $c \in \mathbf{L}\mathscr{S}$ and $\rho \in \mathbf{SUB}$. We first do so for labelled processes. Before proving the relevant theorem we state a technical lemma which we will need in the proof.

LEMMA 4.4. *Let* $c \in \mathbf{L}\mathscr{S}$ *and* $\rho \in \mathbf{SUB}$. *Then* $c\rho\surd$ *and* $(X, c') \in \eta'(c)$ *imply* $\rho(X)\surd$ *and* $c'\rho\surd$.

Proof. By structural induction on c. Omitted. ∎

The next theorem relates the moves of $\pi\rho$, $\pi \in \mathbf{LP}$ and $\rho \in \mathbf{SUB}$, to those of π and ρ. Intuitively it states that if $\pi\rho \stackrel{e}{\Longrightarrow} s$ one of the following situations occur:

- π can execute the start of action a_i, for some $a_i \in \mathbf{LA}$, and the process associated to a_i by ρ can execute the action e. From that moment onwards $F(a_i)$ will be a "place-holder" for what remains to be executed of the process $\rho(a_i)$.

• There is a couple (A, π') associated to π by η such that ρ maps all the elements of A to terminated processes and $\pi'\rho \stackrel{e}{\Longrightarrow} s$. This is justified by Proposition 6.1, which states that, in this case, $\pi\rho$ behaves like $\pi'\rho$ with respect to $\sim_t$.

THEOREM 4.6. *Let* $\pi \in \mathbf{LP}$ *and* $\rho \in \mathbf{SUB}$. *Then* $\pi\rho \stackrel{e}{\Longrightarrow} s$ *implies that*

(i) $\exists a_i : \pi \stackrel{S(a_i)}{\longmapsto} c \wedge \rho(a_i) \stackrel{e}{\Longrightarrow} y \wedge s = c\rho[F(a_i) \to y]$, *or*

(ii) $\exists (X, \pi') \in \eta(\pi) : \rho(X)\surd \wedge \pi'\rho \stackrel{e}{\Longrightarrow} s$.

Proof. By structural induction on π.

$\pi = nil$. Vacuous.

$\pi = a_i$. Clause (i) is trivially met.

$\pi = \pi_1 + \pi_2$. Assume $(\pi_1 + \pi_2)\rho \stackrel{e}{\Longrightarrow} s$. Wlog, we may assume $\pi_1\rho + \pi_2\rho \stackrel{e}{\Longrightarrow} s$ because $\pi_1\rho \stackrel{e}{\Longrightarrow} s$. By inductive hypothesis this implies that

(i) $\exists a_i : \pi_1 \stackrel{S(a_i)}{\longmapsto} c \wedge \rho(a_i) \stackrel{e}{\Longrightarrow} y \wedge s = c\rho[F(a_i) \to y]$, or

(ii) $\exists (X, \pi_1') \in \eta(\pi_1) : \rho(X)\surd \wedge \pi_1'\rho \stackrel{e}{\Longrightarrow} s$.

If (i) holds then

$$\pi_1 + \pi_2 \stackrel{S(a_i)}{\longmapsto} c \wedge \rho(a_i) \stackrel{e}{\Longrightarrow} y \wedge s = c\rho[F(a_i) \to y].$$

If (ii) holds then, by definition of $\eta(\cdot)$, $(X, \pi_1' + \pi_2) \in \eta(\pi_1 + \pi_2)$.
Moreover $\rho(X)\surd$ and $\pi_1'\rho + \pi_2\rho \stackrel{e}{\Longrightarrow} s$.

$\pi = \pi_1 ; \pi_2$. Assume $(\pi_1 ; \pi_2)\rho = \pi_1\rho ; \pi_2\rho \stackrel{e}{\Longrightarrow} s$. There are two possibilities:

(a) $\pi_1\rho \stackrel{e}{\Longrightarrow} s'$ and $s = s' ; \pi_2\rho$. By inductive hypothesis $\pi_1\rho \stackrel{e}{\Longrightarrow} s'$ implies that

(i) $\exists a_i : \pi_1 \stackrel{S(a_i)}{\longmapsto} c \wedge \rho(a_i) \stackrel{e}{\Longrightarrow} y \wedge s' = c\rho[F(a_i) \to y]$, or

(ii) $\exists (X, \pi_1') \in \eta(\pi_1) : \rho(X)\surd$ and $\pi_1'\rho \stackrel{e}{\Longrightarrow} s'$.

If (i) holds then $\pi_1 ; \pi_2 \stackrel{S(a_i)}{\longmapsto} c ; \pi_2$. Moreover

$$(c ; \pi_2)\rho[F(a_i) \to y] = (c\rho[F(a_i) \to y]) ; (\pi_2\rho[F(a_i) \to y]) = s' ; \pi_2\rho.$$

If (ii) holds then, by definition of $\eta(\cdot)$, $(X, \pi_1' ; \pi_2) \in \eta(\pi_1 ; \pi_2)$ and, by the operational semantics,

$$(\pi_1' ; \pi_2)\rho = \pi_1'\rho ; \pi_2\rho \stackrel{e}{\Longrightarrow} s' ; \pi_2\rho.$$

(b) $\pi_1\rho\surd$ and $\pi_2\rho \stackrel{e}{\Longrightarrow} s$. By the inductive hypothesis, $\pi_2\rho \stackrel{e}{\Longrightarrow} s$ implies that

(i) $\exists a_i : \pi_2 \stackrel{S(a_i)}{\longmapsto} c \wedge \rho(a_i) \stackrel{e}{\Longrightarrow} y \wedge s = c\rho[F(a_i) \to y]$, or

(ii) $\exists (X, \pi_2') \in \eta(\pi_2) : \rho(X)\surd \wedge \pi_2'\rho \stackrel{e}{\Longrightarrow} s$.

If (i) holds then there are two subcases:

1. $\pi_1\surd$. Then $\pi_1;\pi_2 \xrightarrow{S(a_i)} c$ and clause (i) is met.

2. $\pi_1 \notin \surd$. Then, by the above lemma, $\pi_1\rho\surd$ implies that for each $(X,\pi_1')\in\eta(\pi_1)$ we have that $\rho(X)\surd$ and $\pi_1'\rho\surd$. Hence, for each $(X,\pi_1';\pi_2)\in\eta(\pi_1;\pi_2)$ we have that $\rho(X)\surd$ and $\pi_1'\rho\surd$. By the operational semantics, for each $(X,\pi_1';\pi_2)\in\eta(\pi_1;\pi_2)$ we have that

$$(\pi_1';\pi_2)\rho=\pi_1'\rho;\pi_2\rho \stackrel{e}{\Longrightarrow} s.$$

If (ii) holds then there are two subcases:

1. If $\pi_1\surd$ then $(X,\pi_2')\in\eta(\pi_1;\pi_2)$ and clause (ii) holds.

2. If $\pi_1\rho\surd$ and $\pi_1\notin\surd$ then, again by the above lemma, for each $(Y,\pi_1')\in\eta(\pi_1)$ we have that $\rho(Y)\surd$ and $\pi_1'\rho\surd$.

Hence, by definition of $\eta(\cdot)$, for each $(Y,\pi_1';\pi_2)\in\eta(\pi_1;\pi_2)$ we have that $\rho(Y)\surd$ and $\pi_1'\rho\surd$. Thus $(Y,\pi_1';\pi_2)\in\eta(\pi_1;\pi_2)$ implies that

$$(\pi_1';\pi_2)\rho=\pi_1'\rho;\pi_2\rho \stackrel{e}{\Longrightarrow} s.$$

$\pi=\pi_1\mid\pi_2$. Assume, wlog, $\pi_1\rho\mid\pi_2\rho \stackrel{e}{\Longrightarrow} s\mid\pi_2\rho$ because $\pi_1\rho\stackrel{e}{\Longrightarrow}s$. By the inductive hypothesis, $\pi_1\rho\stackrel{e}{\Longrightarrow}s$ implies that

(i) $\exists a_i:\pi_1\xrightarrow{S(a_i)}c\wedge\rho(a_i)\stackrel{e}{\Longrightarrow}y\wedge s=c\rho[F(a_i)\to y]$, or

(ii) $\exists(X,\pi_1')\in\eta(\pi_1):\rho(X)\surd\wedge\pi_1'\rho\stackrel{e}{\Longrightarrow}s$.

If (i) holds then $\pi_1\mid\pi_2\xrightarrow{S(a_i)}c\mid\pi_2$. Moreover,

$$(c\mid\pi_2)\,\rho[F(a_i)\to y]=(c\rho[F(a_i)\to y])\mid(\pi_2\rho[F(a_i)\to y])=s\mid\pi_2\rho.$$

If (ii) holds then, by definition of $\eta(\cdot)$, $(X,\pi_1'\mid\pi_2)\in\eta(\pi_1\mid\pi_2)$.
By the operational semantics, $(\pi_1'\mid\pi_2)\rho=\pi_1'\rho\mid\pi_2\rho\stackrel{e}{\Longrightarrow}s\mid\pi_2$.
$\pi=\pi_1\not\mid\pi_2$. Similar to the case above. ∎

To characterize to moves of $c\rho$ for $c\in\mathbf{L}\mathscr{S}$ we need to show how a move of $c\rho$ may depend on a possible move of the state $\rho(F(a_i))$.

THEOREM 4.7. *Let $c\in\mathbf{L}\mathscr{S}$, $\rho\in\mathbf{SUB}$. Assume $c\rho\stackrel{e}{\Longrightarrow}s$. Then*

(i) $\exists a_i\in\mathbf{LA}:c\xrightarrow{S(a_i)}c'\wedge\rho(a_i)\stackrel{e}{\Longrightarrow}y\wedge s=c'\rho[F(a_i)\to y]$, *or*

(ii) $\exists(X,c')\in\eta'(c):\rho(X)\surd\wedge c'\rho\stackrel{e}{\Longrightarrow}s$, *or*

(iii) $\exists a_i\in\mathbf{LA}:\rho(F(a_i))\stackrel{e}{\Longrightarrow}y\wedge s=c\rho[F(a_i)\to y]$ *and $F(a_i)$ occurs in c.*

Proof. By structural induction on c. The details are very similar to those in the proof of the above theorem. ∎

So far we have shown how to relate the moves of $c\rho$, for $c \in \mathbf{L}\mathscr{S}$ and $\rho \in \mathbf{SUB}$, to moves of c and ρ. In what follows we need to be able to go in the opposite direction. Namely, in certain circumstances, given moves of c and moves of ρ, we want to derive the corresponding move of $c\rho$. This information is needed in proving that refine-equivalence is preserved by action-refinement and is derived by the next lemma.

LEMMA 4.5. *Let* $c \in \mathbf{L}\mathscr{S}$, $\rho \in \mathbf{SUB}$.

(a) *Assume* $c \overset{S(a_i)}{\longmapsto} c'$ *and* $\rho(a_i) \overset{e}{\Longrightarrow} y$. *Then* $c\rho \overset{e}{\Longrightarrow} c'\rho[F(a_i) \to y]$.

(b) *Assume* $F(a_i)$ *occurs in* c *and* $\rho(F(a_i)) \overset{e}{\Longrightarrow} y$. *Then* $c\rho \overset{e}{\Longrightarrow} c\rho[F(a_i) \to y]$.

Proof. Both statements can be shown by structural induction on c. We sketch only the proof of statement (a) leaving the proof of (b) to the reader.

$c = \pi$: We show that $\pi \overset{S(a_i)}{\longmapsto} c'$ and $\rho(a_i) \overset{e}{\Longrightarrow} y$ imply $\pi\rho \overset{e}{\Longrightarrow} c'\rho[F(a_i) \to y]$ by induction on the proof of $\pi \overset{S(a_i)}{\longmapsto} c'$.

We examine only the case $\pi = \pi_1 \mid \pi_2$ and leave the others to the reader.

$\pi = \pi_1 \mid \pi_2$. Wlog, assume $\pi_1 \mid \pi_2 \overset{S(a_i)}{\longmapsto} c \mid \pi_2$ because $\pi_1 \overset{S(a_i)}{\longmapsto} c$. By the inductive hypothesis, $\pi_1\rho \overset{e}{\Longrightarrow} c\rho[F(a_i) \to y]$. By the operational semantics,

$$(\pi_1 \mid \pi_2)\rho = \pi_1\rho \mid \pi_2\rho \overset{e}{\Longrightarrow} (c\rho[F(a_i) \to y]) \mid \pi_2\rho.$$

Moreover, as $F(a_i)$ does not occur in π_2,

$$(c \mid \pi_2)\,\rho[F(a_i) \to y] = (c\rho[F(a_i) \to y]) \mid \pi_2\rho.$$

$c = F(a_i)$: Vacuous.

The cases $c = c_1 \mid c_2$ and $c = c_1; \pi$ are easy. ∎

We now have all the technical machinery we need to prove that $\sim_r$ is preserved by action-refinement. The proof is based on the tight correspondence between the places of two configurations c, d such that $c \sim_\phi d$ which is recorded by $\phi \in \mathscr{H}$. This allows us to state formally when a place $F(a_i)$ in c *plays the same role* as a place $F(a_j)$ in d with respect to action-refinement.

DEFINITION 4.10. 1. Let $\rho, \rho' \in \mathbf{SUB}$ and $\phi \in \mathscr{H}$. We say that $\rho \sim_\phi \rho'$ iff for each $a \in \mathbf{A}$

(i) $\forall i \in \mathrm{dom}(\phi_a)\ \rho(F(a_i)) \sim_t \rho'(F(a_{\phi_a(i)}))$

(ii) $\forall i, j \in \mathbf{N}\ \rho(a_i) \sim_t \rho'(a_j)$.

2. Define $\mathscr{R}_{\text{sub}} \subseteq \mathscr{S}^2$ as follows:

$$\mathscr{R}_{\text{sub}} \underset{\text{def}}{=} \{(c\rho, c'\rho') \mid \exists \phi \in \mathscr{H} : c \sim_\phi c' \wedge \rho \sim_\phi \rho'\}.$$

We want to show that $\mathscr{R}_{\text{sub}}$ is a timed-bisimulation. Before showing this result we state a technical lemma which will be useful in its proof.

LEMMA 4.6. (i) *$\mathscr{R}_{\text{sub}}$ is an equivalence relation on $\mathscr{S}$.*

(ii) $\forall s_1, s_2 \in \mathscr{S}\ s_1 \sim_{\text{t}} s_2 \Rightarrow (s_1, s_2) \in \mathscr{R}_{\text{sub}}$.

Proof. (i) This statement is a simple consequence of the properties of $\sim_\phi$ on $\mathbf{L}\mathscr{S}$ and of the following observations:

- $\forall \rho, \rho'\ \rho \sim_\phi \rho' \Rightarrow \rho' \sim_{\phi^{\text{op}}} \rho$.
- $\forall \rho, \rho', \rho''\ \sim_{\phi_1} \rho' \wedge \rho' \sim_{\phi_2} \rho'' \Rightarrow \rho \sim_{\phi_2 \cdot \phi_1} \rho''$.

(ii) Assume that $s_1, s_2 \in \mathscr{S}$ and $s_1 \sim_{\text{t}} s_2$. Then, by Theorem 4.5, $s_1 \sim_{\text{r}} s_2$. This implies that there exist $c_1, c_2 \in \mathbf{L}\mathscr{S}$ such that $\mathbf{un}(c_i) = s_i$, $i = 1, 2$, and $c_1 \sim_\phi c_2$ for some $\phi \in \mathscr{H}$.

Consider $\rho \in \mathbf{SUB}$ such that $\forall a_i \in \mathbf{LA}\ \rho(a_i) = a \wedge \rho(F(a_i)) = F(a)$.

Then it is easy to see that, for each $\phi \in \mathscr{H}$, $\rho \sim_\phi \rho$. By definition of $\mathscr{R}_{\text{sub}}$, $(c_1\rho, c_2\rho) \in \mathscr{R}_{\text{sub}}$. Moreover, $c_i\rho = \mathbf{un}(c_i) = s_i$, $i = 1, 2$. Hence $(s_1, s_2) \in \mathscr{R}_{\text{sub}}$. ∎

THEOREM 4.8. *The relation $\mathscr{R}_{\text{sub}}$ is a timed-bisimulation.*

Proof. We prove by induction on the size of c that for $(c\rho, c'\rho') \in \mathscr{R}_{\text{sub}}$,

$$c\rho \overset{e}{\Longrightarrow} s \Rightarrow \exists s' : c'\rho' \overset{e}{\Longrightarrow} s' \text{ and } (s, s') \in \mathscr{R}_{\text{sub}}.$$

By symmetry this is sufficient to prove the claim.

First of all recall that $(c\rho, c'\rho') \in \mathscr{R}_{\text{sub}}$ iff for some $\phi \in \mathscr{H}\ c \sim_\phi c'$ and $\rho \sim_\phi \rho'$. Assume $c\rho \overset{e}{\Longrightarrow} s$. By Theorem 4.7 there are three cases to examine:

(i) $\exists a_i : c \overset{S(a_i)}{\longmapsto} d \wedge \rho(a_i) \overset{e}{\Longrightarrow} y \wedge s = d\rho[F(a_i) \to y]$. Then, as $c \sim_\phi c'$, there exist $j \in \mathbf{N}$, $d' \in \mathbf{L}\mathscr{S}$ such that

$$d \overset{S(a_j)}{\longmapsto} d' \qquad \text{and} \qquad c' \sim_{\phi \cup \{(a_i, j)\}} d'.$$

As $\rho \sim_\phi \rho'$, $\exists y' : \rho'(a_j) \overset{e}{\Longrightarrow} y'$ and $y \sim_{\text{t}} y'$.

By Lemma 4.5, $d \overset{S(a_j)}{\longmapsto} d'$ and $\rho'(a_j) \overset{e}{\Longrightarrow} y'$ imply

$$d\rho' \overset{e}{\Longrightarrow} d'\rho'[F(a_j) \to y'].$$

It is easy to see that

$$\rho[F(a_i) \to y] \sim_{\phi \cup \{(a_i, j)\}} \rho'[F(a_j) \to y'].$$

Hence, by the definition of $\mathscr{R}_{\text{sub}}$,

$$(d\rho[F(a_i) \to y], d'\rho'[F((a_j) \to y'])\in \mathscr{R}_{\text{sub}}.$$

(ii) $\exists (X, d)\in \eta'(c) : d\rho \stackrel{e}{\Longrightarrow} s$ and $\rho(X)\surd$. Then, by Corollary 4.3, $c\rho \sim_t d\rho$. By the above lemma, $(c\rho, d\rho)\in \mathscr{R}_{\text{sub}}$.

As $\mathscr{R}_{\text{sub}}$ is an equivalence relation, $(d\rho, c'\rho')\in \mathscr{R}_{\text{sub}}$. As the size of d is less than that of c, we may apply induction to obtain

$$\exists s' : c'\rho' \stackrel{e}{\Longrightarrow} s' \text{ and } (s, s')\in \mathscr{R}_{\text{sub}}.$$

(iii) $\exists a_i : \rho(F(a_i)) \stackrel{e}{\Longrightarrow} s \wedge s = c\rho[F(a_i) \to y]$ and $F(a_i)$ occurs in c. As $F(a_i)$ occurs in c and $c \sim_\phi c'$, $i\in \text{dom}(\phi_a)$. By the hypothesis that $\rho \sim_\phi \rho'$, we have that

$$\exists y' : \rho'(F(a_j)) \stackrel{e}{\Longrightarrow} y' \text{ and } y \sim_t y', \text{ where } j = \phi_a(i).$$

As $F(a_j)$ occurs in c' we may apply Lemma 4.5 to obtain

$$c'\rho' \stackrel{e}{\Longrightarrow} c'\rho'[F(a_j) \to y'].$$

As $\rho \sim_\phi \rho'$, we have that $\rho[F(a_i) \to y] \sim_\phi \rho'[F(a_j) \to y']$.

By the definition of $\mathscr{R}_{\text{sub}}$,

$$(c\rho[F(a_i) \to y], c'\rho'[F(a_j) \to y'])\in \mathscr{R}_{\text{sub}}. \quad \blacksquare$$

This theorem allows us to give a behavioural proof of the fact that $\sim_r$ is preserved by action-refinements.

COROLLARY 4.3 (The Refinement Theorem: Behavioural Proof). $\forall p, q, r\in \mathbf{P}\, p \sim_r q$ *implies* $p[a \rightsquigarrow r] \sim_r q[a \rightsquigarrow r]$.

Proof. We show in fact that $p \sim_r q$ implies that $p[a \rightsquigarrow r] \sim_t q[a \rightsquigarrow r]$, which, by Theorem 4.5, is sufficient to prove the claim. Assume $p, q\in \mathbf{P}$ and $p \sim_r q$. Then there exist

$$\pi_1, \pi_2\in \mathbf{LP} : \mathbf{un}(\pi_1) = p, \mathbf{un}(\pi_2) = q \text{ and } \pi_1 \sim_\varnothing \pi_2.$$

Consider the substitution ρ defined as follows:

- $\forall i\in \mathbf{N}\ \rho(a_i) = r$
- $\forall i, b \neq a\ \rho(b_i) = b$
- $\forall i\ \rho(F(a_i)) = F(a)$.

Then $\rho \sim_\varnothing \rho$ and therefore, by the definition of $\mathscr{R}_{\text{sub}}$, $(\pi_1\rho, \pi_2\rho)\in \mathscr{R}_{\text{sub}}$.

Moreover, $\pi_1\rho = p[a/r]$ and $\pi_2\rho = q[a/r]$. By the above theorem and the definition of $\sim_t$ over $\mathbf{P}_{\text{ref}}$, $p[a \rightsquigarrow r] \sim_t q[a \rightsquigarrow r]$. $\blacksquare$

5. Proof of the Equational Characterization

This section is entirely devoted to a detailed proof of the equational characterization of the relation $\sim_t$ over **P** (Theorem 3.2). The proof is based upon standard techniques used in the literature to axiomatize non-interleaving equivalences; see, e.g., [CH89, H88]. However, the details are much more involved because of the presence of general sequential composition in the language, rather than action-prefixing. As usual, the proof of the completeness theorem relies on the isolation of suitable normal forms for processes. For the languages with action-prefixing considered in the above given references, normal forms for processes are of the form

$$\sum_{i \in I} a_i \cdot p_i \mathbin{\vee} q_i,$$

where p_i and q_i are themselves normal forms. In the presence of a general sequential composition operator, however, normal forms will be process terms made up of an arbitrary alternation of sequential composition and left-merge. (See Definition 5.1.) In order to reason about such complex terms, we develop decomposition results for normal forms and their associated process states modulo $\sim_t$. (See Section 5.2.)

5.1. *Preliminaries*

Let $=_E$ denote the least congruence over **P** which satisfies the set of axioms E in Fig. 1. The following proposition, whose proof is standard and thus omitted, states that the axioms are indeed sound with respect to $\sim_t$.

Proposition 5.1 (Soundness). *For each* $p, q \in \mathbf{P}$, $p =_E q$ *implies* $p \sim_t q$.

Note that axiom (C3) is *not* sound without the side-condition. For example, $(nil + b); c \sim_t nil; c + b; c$.

In what follows we concentrate on the much more challenging proof of completeness of the set of axioms E with respect to $\sim_t$. In order to prove the completeness theorem, it is convenient to reduce the processes in **P** to what will be called *head reduced forms* or *hrf*'s. Before giving the definition of this class of terms, let us introduce some useful notation.

Notation 5.1. In what follows, for each process $p \in \mathbf{P}$,

$$p \Rightarrow \text{ iff there exist } a \in \mathbf{A}, s \in \mathcal{S} : p \xRightarrow{S(a)} s.$$

Note that $p \not\Rightarrow$ iff $p \sim_t nil$.

We now introduce a class of processes, the head reduced forms, which is very useful in the proof of the completeness theorem. Intuitively, sequential hrf's correspond to processes which semantically have the sequential com-

position operator as head operator and parallel hrf's to processes which semantically have the left-merge operator as head operator. Combinations of parallel and sequential behaviour in processes are dealt with by means of the larger class of hrf's.

DEFINITION 5.1 (Head Reduced Forms). The sets HRF (head reduced forms), $\mathrm{HRF}_{\mathrm{Seq}}$ (sequential head reduced forms), and $\mathrm{HRF}_{\mathrm{Par}}$ (parallel head reduced forms) are defined simultaneously as the least sets satisfying

- $a; p \in \mathrm{HRF}_{\mathrm{Seq}}$ if $p \in \mathrm{HRF}$;
- $p; q \in \mathrm{HRF}_{\mathrm{Seq}}$ if $p \in \mathrm{HRF}_{\mathrm{Par}}$, $q \in \mathrm{HRF}$, and $q \Rightarrow$;
- $p \between q \in \mathrm{HRF}_{\mathrm{Par}}$ if $p \in \mathrm{HRF}_{\mathrm{Seq}}$, $q \in \mathrm{HRF}$, and $q \Rightarrow$;
- $\sum_{i \in I} p_i \in \mathrm{HRF}$ if I is a finite index set and, for each $i \in I$, $p_i \in \mathrm{HRF}_{\mathrm{Seq}}$ or $p_i \in \mathrm{HRF}_{\mathrm{Par}}$.

By convention, if $I = \varnothing$ then $\sum_{i \in \varnothing} p_i \equiv nil$.

The next lemma states that it is possible to reduce each process in $\mathbf{P}$ to a head reduced form using the axioms in E.

LEMMA 5.1 (Reduction Lemma). *For each $p \in \mathbf{P}$, there exists a head reduced form $h(p)$ such that $p =_E h(p)$.*

Proof. By induction on the depth of p. The proof proceeds by a case analysis on the structure of p and will use all of the axioms in E apart from (A3).

- $p \equiv nil$. Then p is already a hrf.
- $p \equiv a$. Then $a =_E a; nil$, which is a sequential hrf, by axiom (C2).
- $p \equiv q + r$. By the inductive hypothesis, there exist hrf's $h(q)$ and $h(r)$ such that $q =_E h(q)$ and $r =_E h(r)$. If $h(q) \equiv nil$ or $h(r) \equiv nil$ then apply (A2) and (A4) to obtain a hrf. Otherwise $q + r =_E h(q) + h(r)$ which is a head reduced form.
- $p \equiv q; r$. By the inductive hypothesis, $q =_E h(q)$. Assume, wlog, that $h(q) \equiv \sum_{i \in I} q_i$, where each q_i is either a sequential hrf or a parallel hrf.

If $h(q) \equiv nil$ (i.e., $I = \varnothing$) then apply (C2) and the inductive hypothesis to obtain

$$p \equiv q; r =_E nil; h(r) =_E h(r).$$

Otherwise $I \neq \varnothing$ and $p =_E (\sum_{i \in I} q_i)$; $r =_E \sum_{i \in I} q_i; r$, by repeated use of axiom (C3) which is applicable as each q_i is not terminated. We show that, for each $i \in I$, $q_i; r$ may be reduced to either a sequential hrf or a parallel hrf. There are two possibilities to consider:

(1) q_i is a sequential hrf. We distinguish two subcases:

(i) $q_i \equiv a; \bar{q}$ with $\bar{q} \in \mathrm{HRF}$. Then

$$q_i; r \equiv (a; \bar{q}); r =_E a; (\bar{q}; r) \qquad \text{by (C1)}$$
$$=_E a; h(\bar{q}; r) \qquad \text{by the inductive hypothesis.}$$

(ii) $q_i \equiv \hat{q}_1; \hat{q}_2$ with $\hat{q}_1 \in \mathrm{HRF}_{\mathrm{Par}}$, $\hat{q}_2 \in \mathrm{HRF}$, and $\hat{q}_2 \Rightarrow$. Then

$$q_i; r \equiv (\hat{q}_1; \hat{q}_2); r =_E \hat{q}_1; (\hat{q}_2; r) \qquad \text{by (C1)}$$
$$=_E \hat{q}_1; h(\hat{q}_2; r) \qquad \text{by the inductive hypothesis,}$$

which is a sequential hrf as $h(\hat{q}_2; r) \Rightarrow$.

(2) q_i is a parallel hrf. Then $q_i \equiv \hat{q}_1 \between \hat{q}_2$ with $\hat{q}_1 \in \mathrm{HRF}_{\mathrm{Seq}}$, and $\hat{q}_2 \Rightarrow$. If $h(r) \equiv nil$ then $q_i; r =_E q_i$. Otherwise, $h(r) \Rightarrow$ and $q_i; r \equiv (\hat{q}_1 \between \hat{q}_2); h(r)$, which is a sequential hrf.

• $p \equiv q \between r$. By the inductive hypothesis, $q =_E h(q)$. Assume, wlog, that $h(q) \equiv \sum_{i \in I} q_i$. If $h(q) \equiv nil$ then apply (B4) to obtain $p =_E nil$. Otherwise, $p =_E (\sum_{i \in I} q_i) \between r =_E \sum_{i \in I} q_i \between r$ by repeated use of (B1).

We show that, for each $i \in I$, $q_i \between r$ may be reduced to either a sequential hrf or a parallel hrf. There are two cases to examine:

(1) $q_i \in \mathrm{HRF}_{\mathrm{Seq}}$. We distinguish two subcases:

(i) $q_i \equiv a; \bar{q}$. Then $a; \bar{q} \between r =_E a; \bar{q} \between h(r)$ by the induction hypothesis. If $h(r) \equiv nil$ then apply (B3) to obtain a sequential hrf. Otherwise $h(r) \Rightarrow$ and $a; \bar{q} \between h(r)$ is a parallel hrf.

(ii) $q_i \equiv \hat{q}_1; \hat{q}_2$ with $\hat{q}_1 \in \mathrm{HRF}_{\mathrm{Par}}$, $\hat{q}_2 \in \mathrm{HRF}$ and $\hat{q}_2 \Rightarrow$. Then

$$q_i \between r =_E (\hat{q}_1; \hat{q}_2) \between h(r) \qquad \text{by the inductive hypothesis}$$
$$=_E \begin{cases} \hat{q}_1; \hat{q}_2 \in \mathrm{HRF}_{\mathrm{Seq}} & \text{if } h(r) \equiv nil \\ (\hat{q}_1; \hat{q}_2) \between h(r) & \text{otherwise} \end{cases}$$

(2) $q_i \equiv \hat{q}_1 \between \hat{q}_2$ with $\hat{q}_1 \in \mathrm{HRF}_{\mathrm{Seq}}$, $\hat{q}_2 \in \mathrm{HRF}$ and $\hat{q}_2 \Rightarrow$. Then

$$q_i \between r \equiv (\hat{q}_1 \between \hat{q}_2) \between r =_E \hat{q}_1 \between (\hat{q}_2 \mid r) \qquad \text{by (B2)}$$
$$=_E \hat{q}_1 \between h(\hat{q}_2 \mid r) \qquad \text{by the inductive hypothesis,}$$

which is a parallel hrf as $\hat{q}_2 \Rightarrow$ implies $h(\hat{q}_2 \mid r) \Rightarrow$.

• $p \equiv q \mid r$. Then $p =_E q \between r + r \between q$ by (X1). The claim now follows from the above case.

This completes the proof of the reduction lemma. ∎

The proof of the completeness theorem is based upon several lemmas stating important decomposition properties. As we are dealing with finite processes, it will be convenient to prove most of these properties by induction on some notion of the size of a process. In what follows, the *size* of a process p, $|p|$, will be taken to be the length of the longest sequence of subactions it can perform with respect to the timed operational semantics, i.e.,

$$|p| \underset{\text{def}}{=} \max\{l(\sigma) \mid \sigma \in \mathbf{A}_s^* \text{ and } p \stackrel{\sigma}{\Longrightarrow} s, \text{ for some } s\},$$

where $l(\sigma)$ denotes the length of the sequence σ. This notion of size is generalized in this obvious way to $s \in \mathscr{S}$. The following proposition is easily established.

PROPOSITION 5.2. *For each* $s_1, s_2 \in \mathscr{S}$, $s_1 \sim_t s_2$ *implies* $|s_1| = |s_2|$.

LEMMA 5.2. *Let* $s_1, s_2 \in \mathscr{S}$. *Then* $s_1; p \sim_t s_2; p$ *implies* $s_1 \sim_t s_2$.

Proof. By induction on the combined size of s_1 and s_2.

• Suppose $s_1\surd$. Assume, towards a contradiction, that there exist $e \in \mathbf{A}_s$, s_2' such that $s_2 \stackrel{e}{\Longrightarrow} s_2'$. Then, by the operational semantics, $s_2; p \stackrel{e}{\Longrightarrow} s_2'; p$. As $s_1; p \sim_t s_2; p$, there exists s such that $p \stackrel{e}{\Longrightarrow} s$ and $s \sim_t s_2'; p$. However, this is impossible as $|s| < |p| \leqslant |s_2'; p|$. Thus $s_2\surd$ and therefore $s_1 \sim_t s_2$.

• Assume $s_1 \stackrel{e}{\Longrightarrow} s_1'$. By the operational semantics, this implies that $s_1; p \stackrel{e}{\Longrightarrow} s_1'; p$. Reasoning as above we may deduce that *not* $s_2\surd$. Hence there exists s_2' such that $s_2 \stackrel{e}{\Longrightarrow} s_2'$ and $s_1'; p \sim_t s_2'; p$. By the inductive hypothesis, $s_1' \sim_t s_2'$. As this can be done for each e, s_1' such that $s_1 \stackrel{e}{\Longrightarrow} s_1'$, we obtain, by symmetry, that $s_1 \sim_t s_2$. ▮

Two special subclasses of the set of states $\mathscr{S}$ play an important role in the proof of the completeness theorem. These are called sequential configurations and parallel configurations. Intuitively, sequential configurations are, as their name indicates, semantically sequential process states, in a sense which is made precise in the following section. (See Definition 5.4 and Proposition 5.5.) Dually, parallel configurations are process states which are semantically parallel, in a sense which is formalized in Definition 5.3. These two sets of states are also closely related to the classes of sequential and parallel hrf's and help us to obtain a precise understanding of the behaviour of these processes. In fact, we show that, modulo $\sim_t$, the target states of start-moves performed by sequential hrf's are sequential configurations. Again, a dual result holds for parallel hrf's: the target states of start-moves performed by these processes are parallel hrf's. (See Lemma 5.3.)

Notation 5.2. For each index set $I=\{i_1, ..., i_k\}$, $k \geq 0$, the notation $\prod_{i \in I} c_i$ stands for $c_{i_1} \mid \cdots \mid c_{i_k}$ and is justified by commutativity and associativity of $\mid$ modulo $\sim_t$. By convention, $\prod_{i \in \varnothing} c_i \equiv nil$.

DEFINITION 5.2 (Parallel and Sequential Configurations). The sets $\mathscr{C}_{\text{Seq}}$, the set of *sequential configurations*, and $\mathscr{C}_{\text{Par}}$, the set of *parallel configurations*, are simultaneously defined as the least subsets of $\mathscr{S}$ which satisfy the following clauses:

- $F(a)$; $p \in \mathscr{C}_{\text{Seq}}$ if $p \in \mathbf{P}$;
- $c \in \mathscr{C}_{\text{Par}}$, $p \Rightarrow$ imply c; $p \in \mathscr{C}_{\text{Seq}}$;
- $\{c_i \mid i \in I\} \subseteq \mathscr{C}_{\text{Seq}}$, $|I| > 0$ and $p \Rightarrow$ imply $(\prod_{i \in I} c_i) \mid p \in \mathscr{C}_{\text{Par}}$;
- $\{c_i \mid i \in I\} \subseteq \mathscr{C}_{\text{Seq}}$, $|I| > 1$ imply $\prod_{i \in I} c_i \in \mathscr{C}_{\text{Par}}$.

In the above chauses I is always assumed to stand for a *finite* index set.

The following immediate properties of sequential and parallel configurations are heavily used in the proofs of the main results of this section.

Fact 5.1. (1) For each $c \in \mathscr{C}_{\text{Seq}} \cup \mathscr{C}_{\text{Par}}$, there exists $a \in \mathbf{A}$ such that $c \xrightarrow{F(a)}$.

(2) For each $c \in \mathscr{C}_{\text{Par}}$,

(a) $c \xrightarrow{S(a)}$, for some $a \in \mathbf{A}$, or

(b) $c \xrightarrow{F(a)} \xrightarrow{F(b)}$, for some $a, b \in \mathbf{A}$.

Hence sequential configurations are always capable of performing the end of at least one action; on the other hand, parallel configurations are either capable of performing the start of an action or can perform at least two end moves in a row. The following lemmas study the effect of start- and end-moves on processes and configurations. At this stage it is useful to introduce the notation $c \in \sim_t X$ to mean that there exists some $x \in X$ such that $c \sim_t x$.

PROPOSITION 5.3. (i) *For each* $s \in \mathscr{S}$, $s \in \mathbf{P}$ *or* $s \in \sim_t \mathscr{C}_{\text{Seq}}$ *or* $s \in \sim_t \mathscr{C}_{\text{Par}}$.

(ii) *Let* $s_1, s_2 \in \mathscr{S}$. *Assume that* $|s_1|, |s_2| > 0$. *Then* $s_1 \mid s_2 \in \mathbf{P}$ *or* $s_1 \mid s_2 \in \sim_t \mathscr{C}_{\text{Par}}$.

Proof. (i) By induction on the structure of s. We only sketch the proof of the case $s = s_1 \mid s_2$. In this case, by the inductive hypothesis, we have that, for $i = 1, 2$,

$$s_i \in \mathbf{P} \quad \text{or} \quad s_i \in \sim_t \mathscr{C}_{\text{Seq}} \quad \text{or} \quad s_i \in \sim_t \mathscr{C}_{\text{Par}}.$$

If $s_1, s_2 \in \mathbf{P}$ then $s_1 \mid s_2 \in \mathbf{P}$. If $|s_1| = 0$ and $s_2 \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ or $s_1 \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ and $|s_2| = 0$ then $s_2 \mid s_1 \in \sim_t \mathscr{C}_{\mathrm{Seq}}$. In all the other possible cases we have that $s_1 \mid s_2 \in \sim_t \mathscr{C}_{\mathrm{Par}}$.

(ii) Follows from the previous analysis for (i). ∎

The following lemma studies the effect of start-moves on sequential and parallel head reduced forms.

LEMMA 5.3. (1) *$p \in \mathrm{HRF}_{\mathrm{Seq}}$ and $p \xrightarrow{S(a)} c$ imply $c \in \mathscr{C}_{\mathrm{Seq}}$.*

(2) *$p \in \mathrm{HRF}_{\mathrm{Par}}$ and $p \xrightarrow{S(a)} c$ imply $c \in \mathscr{C}_{\mathrm{Par}}$ and c is of the form $d \mid r$, with $d \in \mathscr{C}_{\mathrm{Seq}}$, $r \in \mathbf{P}$ and $|r| > 0$.*

Proof. Both statements are shown by simultaneous induction on the size of p. We assume, as inductive hypothesis, that (1) and (2) hold for each q such that $|q| < |p|$.

(1) Assume $p \in \mathrm{HRF}_{\mathrm{Seq}}$ and $p \xrightarrow{S(a)} c$. By the definition of the set $\mathrm{HRF}_{\mathrm{Seq}}$ there are two cases to examine.

- p is of the form $a; q$. Then, by the operational semantics, $c \equiv F(a); q \in \mathscr{C}_{\mathrm{Seq}}$.
- p is of the form $q; r$, with $q \in \mathrm{HRF}_{\mathrm{Par}}$, $r \in \mathrm{HRF}$ and $|r| > 0$. By the operational semantics and the fact that $q \in \mathrm{HRF}_{\mathrm{Par}}$ implies $q \notin \surd$, $q; r \xrightarrow{S(a)} c$ implies $c \equiv d; r$, with $q \xrightarrow{S(a)} d$. As $|r| > 0$, we have that $|q| < |p|$. Thus we may apply the inductive hypothesis for (2) to obtain, among other things, that $d \in \mathscr{C}_{\mathrm{Par}}$. Hence $d; r \in \mathscr{C}_{\mathrm{Seq}}$.

This completes the proof of (1).

(2) Assume $p \in \mathrm{HRF}_{\mathrm{Par}}$ and $p \xrightarrow{S(a)} c$. Then, by the definition of $\mathrm{HRF}_{\mathrm{Par}}$, p is of the form $q \between r$, with $q \in \mathrm{HRF}_{\mathrm{Seq}}$, $r \in \mathrm{HRF}$ and $|r| > 0$. By the operational semantics, $q \between r \xrightarrow{S(a)} c$ implies $c \equiv d \mid r$, with $q \xrightarrow{S(a)} d$. As $|r| > 0$, $|q| < |p|$. We may thus apply the inductive hypothesis for (1) to obtain that $d \in \mathscr{C}_{\mathrm{Seq}}$. Hence $d \mid r \in \mathscr{C}_{\mathrm{Par}}$ and is of the required form. This completes the proof of (2). ∎

The following result is an immediate corollary of the previous lemma.

COROLLARY 5.1 (Effect of Start-Moves on Head Reduced Forms). *Let $p \in \mathrm{HRF}$. Then $p \xrightarrow{S(a)} c$ implies $c \in \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$. Moreover, if $c \in \mathscr{C}_{\mathrm{Par}}$ then c is of the form $d \mid q$, with $d \in \mathscr{C}_{\mathrm{Seq}}$ and $|q| > 0$.*

LEMMA 5.4 (Effect of Start-Moves on Processes). *If $p \in \mathbf{P}$ and $p \xrightarrow{S(a)} c$ then $c \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ or $c \in \sim_t \mathscr{C}_{\mathrm{Par}}$. Moreover, if $c \in \sim_t \mathscr{C}_{\mathrm{Par}}$ then $c \sim_t d \mid q$, for some d, q such that $d \in \mathscr{C}_{\mathrm{Seq}}$ and $|q| > 0$.*

Proof. Let $p \in \mathbf{P}$ and assume that $p \xRightarrow{S(a)} c$. By Lemma 5.1, $p =_E h(p)$ for some hrf $h(p)$. By the soundness of $=_E$ with respect to $\sim_t$, we have that $p \sim_t h(p)$. Hence there exists d such that $h(p) \xRightarrow{S(a)} d \sim_t c$. By the previous corollary, $d \in \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$. Thus $c \in \sim_t \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$. Moreover, again by the previous corollary, $c \in \sim_t \mathscr{C}_{\mathrm{Par}}$ implies that d is of the required form. ∎

LEMMA 5.5 (Effect of Start-Moves on Configurations). *For each* $c \in \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$ *the following statements hold*:

(1) $c \in \mathscr{C}_{\mathrm{Seq}}$ *and* $c \xRightarrow{S(a)} c'$ *imply* $c' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$;

(2) $c \in \mathscr{C}_{\mathrm{Par}}$ *and* $c \xRightarrow{S(a)} c'$ *imply* $c' \in \sim_t \mathscr{C}_{\mathrm{Par}}$.

Proof. Both statements are proved simultaneously by induction on the size of c. We assume, as inductive hypothesis, that (1) and (2) are true of all d with $|d| < |c|$.

(1) The statement is vacuously true if c is of the form $F(a); p$. Assume now c is of the form $d; p$ with $d \in \mathscr{C}_{\mathrm{Par}}$ and $p \Rightarrow$. Then $d; p \xRightarrow{S(a)} d'; p$ if $d \xRightarrow{S(a)} d'$, as for no $d \in \mathscr{C}_{\mathrm{Par}}$, $d\surd$. By the inductive hypothesis for (2) we have that $d' \in \sim_t \mathscr{C}_{\mathrm{Par}}$. Thus $d'; p \in \sim_t \mathscr{C}_{\mathrm{Seq}}$.

(2) Assume c is of the form $\prod_{i \in I} c_i \mid p$, with $|I| > 0$ and $p \Rightarrow$. If $\prod_{i \in I} c_i \mid p \xRightarrow{S(a)} c'$ there are two cases to examine:

- suppose, wlog, that $\prod_{i \in I} c_i \mid p \xRightarrow{S(a)} c_1' \mid \prod_{i \neq 1} c_i \mid p$ because $c_1 \xRightarrow{S(a)} c_1'$. As $c_1 \in \mathscr{C}_{\mathrm{Seq}}$ we may apply the inductive hypothesis for (1) to obtain $c_1' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$. Hence $c_1' \mid \prod_{i \neq 1} c_i \mid p \in \sim_t \mathscr{C}_{\mathrm{Par}}$;

- suppose $\prod_{i \in I} c_i \mid p \xRightarrow{S(a)} \prod_{i \in I} c_i \mid d$ because $p \xRightarrow{S(a)} d$. Then, by Lemma 5.4, $d \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ or $d \in \sim_t \mathscr{C}_{\mathrm{Par}}$. In both cases $\prod_{i \in I} c_i \mid d \in \sim_t \mathscr{C}_{\mathrm{Par}}$.

If c is of the form $\prod_{i \in I} c_i$, $|I| > 1$, then, wlog, $\prod_{i \in I} c_i \xRightarrow{S(a)} c_1' \mid \prod_{i \neq 1} c_i$ because $c_1 \xRightarrow{S(a)} c_1'$. As $c_1 \in \mathscr{C}_{\mathrm{Seq}}$ we may apply the inductive hypothesis for (1) to obtain $c_1' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$. Thus $c_1' \mid \prod_{i \neq 1} c_i \in \sim_t \mathscr{C}_{\mathrm{Par}}$.

This completes the proof of the lemma. ∎

By the above lemmas, we may assume from now on that, modulo $\sim_t$, $\mathscr{C}_{\mathrm{Seq}}$ and $\mathscr{C}_{\mathrm{Par}}$ are closed with respect to moves of the form $\xRightarrow{S(a)}$, $a \in \mathbf{A}$. The following lemma studies the effect of end-moves on configurations.

LEMMA 5.6 (Effect of End-Moves on Configurations). *For each* $c \in \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$, *the following* statements hold:

(1) $c \in \mathscr{C}_{\mathrm{Seq}}$ *and* $c \xRightarrow{F(a)} c'$ *imply* $c' \in \mathbf{P}$ *or* $c' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$;

(2) $c \in \mathscr{C}_{\mathrm{Par}}$ *and* $c \xRightarrow{F(a)} c'$ *imply* $c' \in \mathbf{P}$ *or* $c' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ *or* $c' \in \sim_t \mathscr{C}_{\mathrm{Par}}$.

Proof. Both statements are proved simultaneously by induction on the size of c. We assume, as inductive hypothesis, that (1) and (2) are true for all d with $|d| < |c|$.

(1) If c is of the form $F(a); p$ then $F(a); p \xrightarrow{F(a)} nil; p \in \mathbf{P}$. Otherwise c is of the form $d; p$ with $d \in \mathscr{C}_{\mathrm{Par}}$ and $p \not\Rightarrow$. Assume that $d; p \xrightarrow{F(a)} d'; p$ because $d \xrightarrow{F(a)} d'$. Then, by the inductive hypothesis for (2), $d' \in \mathbf{P}$ or $d' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ or $d' \in \sim_t \mathscr{C}_{\mathrm{Par}}$. If $d' \in \mathbf{P}$ then $d'; p \in \mathbf{P}$. If $d' \in \sim_t \mathscr{C}_{\mathrm{Par}}$ then $d'; p \in \sim_t \mathscr{C}_{\mathrm{Seq}}$. Otherwise $d' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ and there are two possibilities to consider:

- $d' \sim_t F(b); q$. Then $d'; p \sim_t F(b); (q; p) \in \mathscr{C}_{\mathrm{Seq}}$.
- $d' \sim_t d''; q$ with $d'' \in \mathscr{C}_{\mathrm{Par}}$ and $q \not\Rightarrow$. Then $d'; p \sim_t d''; (q; p) \in \mathscr{C}_{\mathrm{Seq}}$.

This completes the proof of (1).

(2) Assume $c \in \mathscr{C}_{\mathrm{Par}}$. We distinguish two cases:

- c is of the form $\prod_{i \in I} c_i \mid p$, $|I| > 0$ and $p \not\Rightarrow$. Then, wlog, $\prod_{i \in I} c_i \mid p \xrightarrow{F(a)} c_1' \mid \prod_{i \neq 1} c_i \mid p$ because $c_1 \xrightarrow{F(a)} c_1'$. By the inductive hypothesis for (1), $c_1' \in \mathbf{P}$ or $c_1' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$. If $c_1' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ then $c_1' \mid \prod_{i \neq 1} c_i \mid p \in \sim_t \mathscr{C}_{\mathrm{Par}}$. Otherwise $c_1' \in \mathbf{P}$. If $|I| = 1$ then $c_1' \mid \prod_{i \neq 1} c_i \mid p \in \mathbf{P}$. If $|I| > 1$ then $c_1' \mid \prod_{i \neq 1} c_i \mid p \in \sim_t \mathscr{C}_{\mathrm{Par}}$ as $c_1' \mid p \not\Rightarrow$.
- c is of the form $\prod_{i \in I} c_i$, $|I| > 1$. Assume, wlog, that $\prod_{i \in I} c_i \xrightarrow{F(a)} c_1' \mid \prod_{i \neq 1} c_i \equiv c'$ because $c_1 \xrightarrow{F(a)} c_1'$. By the inductive hypothesis for (1), $c_1' \in \mathbf{P}$ or $c_1' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$. If $c_1' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$ then $c' \in \sim_t \mathscr{C}_{\mathrm{Par}}$. Otherwise $c_1' \in \mathbf{P}$. Now, if $|I| > 2$ then $c' \in \sim_t \mathscr{C}_{\mathrm{Par}}$. If $|I| = 2$ and $c_1' \not\Rightarrow$ then $c' \in \sim_t \mathscr{C}_{\mathrm{Par}}$. If $|I| = 2$ and $c_1' \sim_t nil$ then $c' \in \sim_t \mathscr{C}_{\mathrm{Seq}}$.

This completes the proof of (2). ∎

The following lemma will find application in some of the results presented in the following section.

LEMMA 5.7. *Let $c \in \mathscr{C}_{\mathrm{Par}}$. Assume that*

- $c \xrightarrow{S(a)}$, *for each* $a \in \mathbf{A}$, *and*
- *for no* $a \in \mathbf{A}$, $d \in \sim_t \mathscr{C}_{\mathrm{Par}}$, $c \xrightarrow{F(a)} d$.

Then $c \sim_t F(a) \mid F(b)$, *for some* $a, b \in \mathbf{A}$.

Proof. First of all, note that, by the proviso of the lemma, c must be of the form $c_1 \mid c_2$ with $c_1, c_2 \in \mathscr{C}_{\mathrm{Seq}}$. We claim that $c_1 \sim_t F(a)$ and $c_2 \sim_t F(b)$, for some $a, b \in \mathbf{A}$. Assume in fact, for the sake of contradiction, that $c_1 \in \mathscr{C}_{\mathrm{Seq}}$ and, for no $a \in \mathbf{A}$, $c_1 \sim_t F(a)$. Then either $c_1 \sim_t F(a); p$, for some $a \in \mathbf{A}$ and p such that $|p| > 0$, or $c_1 \sim_t d; p$, for some $d \in \mathscr{C}_{\mathrm{Par}}$ and p such that $|p| > 0$. In both cases, it is easy to see that, by the previous lemma,

$c_1 \mid c_2 \xRightarrow{F(a)} c_1' \mid c_2 \in \sim_t \mathscr{C}_{\mathrm{Par}}$, for some a and c_1'. This contradicts the hypotheses of the lemma. Hence, by symmetry, $c_1 \sim_t F(a)$ and $c_2 \sim_t F(b)$, for some a, b. ∎

The previous lemmas state the basic operational material which finds extensive application in the proof of results leading to the promised completeness theorem. Our next aim is to prove two fundamental decomposition properties used in the proof of the completeness theorem: the sequential and the parallel decomposition theorems. The proofs of these results makes use of some general unique factorization results with respect to $\sim_t$, which are based upon similar results presented in [Mol89, MM90], for $\sim$. The proofs of the factorization and decomposition results are the subject of the following section. In what follows rather than always working modulo $\sim_t$ we reduce the relation $\in \sim_t$ to $\in$; i.e., we assume that $\in$ is always considered modulo $\sim_t$.

5.2. *Unique Factorization and Decomposition Results*

This section will be entirely devoted to proving two fundamental decomposition results which will find application in the proof of the completeness theorem: the sequential decomposition and the parallel decomposition theorems. The sequential decomposition result we are after states that, for each $c, d \in \mathscr{C}_{\mathrm{Par}}$ and $p, q \in \mathbf{P}$,

$$c; p \sim_t d; q \qquad \text{implies} \qquad c \sim_t d \text{ and } p \sim_t q. \tag{1}$$

The parallel decomposition theorem we should like to prove, which may be seen as the dual statement of (1), states that, for each $c, d \in \mathscr{C}_{\mathrm{Seq}}$ and $p, q \in \mathbf{P}$,

$$c \mid p \sim_t d \mid q \qquad \text{implies} \qquad c \sim_t d \text{ and } p \sim_t q. \tag{2}$$

In the process of proving (1) and (2) we shall establish unique factorization results, modulo $\sim_t$, for states $s \in \mathscr{S}$ with respect to the operators of sequential and parallel composition, which are similar to the ones presented in [Mol89, MM90] modulo strong bisimulation equivalence, $\sim$. Apart from their intrinsic interest, such factorization results allow us to give elegant proofs of the above-mentioned decomposition results. The theory presented in this section also allows us to shed more light on the semantic properties of $\sim_t$. In particular, by using the factorization and decomposition results stated above we shall prove that, modulo $\sim_t$, each process $p \in \mathbf{P}$ is either *nil* or semantically sequential or semantically parallel. (See Theorem 5.5 and the subsequent corollary.) This property does not hold for standard bisimulation equivalence, $\sim$, over $\mathbf{P}$.

Our first aim in this section is to show the sequential decomposition result stated in (1). As mentioned above, the proof of (1) relies on a unique

sequential factorization result, which states that each state $s \in \mathscr{S}$ may be expressed uniquely, modulo $\sim_t$, as a sequential composition of "semantically parallel states." Intuitively, a state s is "semantically parallel" if it cannot be expressed, modulo $\sim_t$, as a nontrivial sequential composition of states. Formally:

DEFINITION 5.3 (Seq-irreducible and Seq-prime States). Let $s \in \mathscr{S}$. Then s is said to be:

(i) *seq-irreducible* if $s \sim_t c; p$, for some $c \in \mathscr{S}$ and $p \in \mathbf{P}$, implies $c \sim_t nil$ or $p \sim_t nil$;

(ii) *seq-prime* if s is seq-irreducible and $s \not\sim_t nil$.

For each $s \in \mathscr{S}$, a *sequential factorization* for s modulo $\sim_t$ is a sequence of seq-primes $s_1, p_2, ..., p_n$, with $n \geqslant 0$, $s_1 \in \mathscr{S}$ and, for each $2 \leqslant i \leqslant n$, $p_i \in \mathbf{P}$, such that

$$s \sim_t s_1; p_2; ...; p_n.$$

By convention, if $n = 0$ then $s_1; p_2; ...; p_n \equiv nil$. We shall now prove that each $s \in \mathscr{S}$ has a *unique* sequential factorization into seq-primes, modulo $\sim_t$.

THEOREM 5.1 (Unique Sequential Factorization). *Let $s \in \mathscr{S}$. Then s has a unique sequential factorization into seq-primes modulo $\sim_t$.*

Proof. The proof is divided into two parts, an existence part and a uniqueness part, both of which are shown by induction on $|s|$, the size of $s \in \mathscr{S}$.

Existence. We assume, as inductive hypothesis, that each s' with $|s'| < |s|$ has a sequential factorization. The proof proceeds by a case analysis on the possible form s may take, modulo $\sim_t$. If $s \sim_t nil$ then s has an empty factorization into seq-primes. If s is seq-prime then it is its own factorization. Otherwise we may assume, wlog, that $s \sim_t s_1; p$ with $|s_1|, |p| > 0$. By the inductive hypothesis, $c_1; q_2; ...; q_n$ and $p_1; ...; p_m$ are seq-prime factorizations for s_1 and p, respectively. Thus, by substitutivity,

$$s \sim_t c_1; q_2; ...; q_n; p_1; ...; p_m$$

is a seq-prime factorization for s. This completes the proof of the existence part of the result.

Uniqueness. We assume, as inductive hypothesis, that each $s' \in \mathscr{S}$ such that $|s'| < |s|$ has a unique seq-prime factoriation. Assume that

$$s \sim_t c_1; p_2; ...; p_n$$
$$\sim_t d_1; q_2; ...; q_m$$

are two factorizations of s into seq-primes. We shall show that the two factorizations are identical, modulo $\sim_t$. If $n=0$ then $m=0$ and we are done. If $n=1$ then s is a seq-prime and thus $m=1$ and $c_1 \sim_t d_1$. Hence the two factorizations are identical. Otherwise, by symmetry, we may assume that $n, m \geqslant 2$. We distinguish two possibilities:

(i) $p_n \sim_t q_m$, and

(ii) $p_n \not\sim_t q_m$.

We examine the two possibilities in turn.

(i) Assume that $p_n \sim_t q_m$. Then, by Lemma 5.2 and substitutivity, we have that

$$c_1; p_2; ...; p_{n-1} \sim_t d_1; q_2; ...; q_{m-1}. \tag{3}$$

As $|p_n|, |q_m| > 0$, we may apply the inductive hypothesis to (3) to obtain that $c_1, p_2, ..., p_{n-1}$ and $d_1, q_2, ..., q_{m-1}$ are identical seq-prime factorizations. Thus we have that $n=m$, $c_1 \sim_t d_1$ and, for each $1 < i \leqslant n$, $p_i \sim_t q_i$.

(ii) Assume now, for the sake of contradiction, that $p_n \not\sim_t q_m$. As $|c_1| > 0$, we have that $c_1 \stackrel{e}{\Longrightarrow} c_1'$ for some $e \in \mathbf{A}_s$ and c_1'. By the operational semantics, $c_1; p_2; ...; p_n \stackrel{e}{\Longrightarrow} c_1'; p_2; ...; p_n$. Moreover, as $|d_1| > 0$ and $c_1; p_2; ...; p_n \sim_t d_1; q_2; ...; q_n$, we have that, for some d_1',

$$d_1; q_2; ...; q_m \stackrel{e}{\Longrightarrow} d_1'; q_2; ...; q_m \sim_t c_1'; p_2; ...; p_n.$$

By the inductive hypothesis,

$$\begin{aligned} \bar{s} &\sim_t c_1'; p_2; ...; p_m \\ &\sim_t d_1'; q_2; ...; q_m \end{aligned}$$

has a unique seq-prime factorization

$$c^1; ...; c^h; p_1; ...; p_m \sim_t d^1; ...; d^k; q_2; ...; q_m,$$

where $h, k \geqslant 0$ and $c_1' \sim_t c^1; ...; c^h$ and $d_1' \sim_t d^1; ...; d^k$ are the unique seq-prime factorizations for c_1' and d_1', respectively. As $n, m \geqslant 2$, we then have that $p_n \sim_t q_m$. This contradicts the hypothesis that $p_n \not\sim_t q_m$.

This completes the proof of the uniqueness part and that of the theorem. ∎

We now show the above-given unique sequential factorization result for states, modulo $\sim_t$, may be used to give a proof of the sequential decomposition theorem. First of all, we show that each parallel configuration c is indeed a seq-prime state. This result is a corollary of the following useful proposition.

PROPOSITION 5.4. *Let* $c \in \mathscr{C}_{\mathrm{Par}}$, $d \in \mathscr{S}$, *and* $p \in \mathbf{P}$. *Assume that* $c \sim_t d; p$. *Then* $p \sim_t nil$.

Proof. By induction on the size of c. Assume that $c \sim_t d; p$. Note that, as $c \in \mathscr{C}_{\mathrm{Par}}$ implies that $c \stackrel{F(a)}{\Longrightarrow}$ for some a, this implies that $|d| > 0$. Thus each initial move from $d; p$ has to come from d. The proof proceeds by an analysis of the following three possibilities:

(a) $c \stackrel{S(a)}{\Longrightarrow} c'$, for some a, c',

(b) $c \stackrel{F(a)}{\Longrightarrow} c' \in \mathscr{C}_{\mathrm{Par}}$, for some a, c', and

(c) neither of the two previous cases applies.

We examine each possibility in turn.

(a) In this case, by Lemma 5.5, $c' \in \mathscr{C}_{\mathrm{Par}}$. Moreover, as $c \sim_t d; p$, $d; p \stackrel{S(a)}{\Longrightarrow} d'; p \sim_t c'$ for some d'. We may now apply the inductive hypothesis to obtain $p \sim_t nil$.

(b) Similar to (a).

(c) Assume now that neither of the two previous cases holds. We are then in a position to apply Lemma 5.3 to obtain that $c \sim_t F(a) \mid F(b)$, for some a, b. Then, as $c \sim_t d; p$ and $c \stackrel{F(a)}{\Longrightarrow} \stackrel{F(b)}{\Longrightarrow} nil \mid nil$, there exists d' such that $d; p \stackrel{F(a)}{\Longrightarrow} \stackrel{F(b)}{\Longrightarrow} d'; p \sim_t nil \mid nil$. This implies that $d' \sim_t p \sim_t nil$.

This completes the proof of the proposition. ∎

COROLLARY 5.2. *Each parallel configuration is seq-prime.*

Proof. Assume that $c \in \mathscr{C}_{\mathrm{Par}}$ and that $c \sim_t d; p$. Then, by the above proposition, we have that $p \sim_t nil$. Thus c is seq-irreducible. Moreover, as $c \in \mathscr{C}_{\mathrm{Par}}$ implies $|c| > 0$, c is seq-prime. ∎

We can now prove the promised sequential decomposition theorem.

THEOREM 5.2 (Sequential Decomposition Theorem). *Let* $c, d \in \mathscr{C}_{\mathrm{Par}}$. *Then* $c; p \sim_t d; q$ *implies* $c \sim_t d$ *and* $p \sim_t q$.

Proof. By Theorem 5.1, p and q have unique factorizations into seq-primes given by

$$p \sim_t p_1; \ldots; p_n \qquad \text{and} \qquad q \sim_t q_1; \ldots; q_m.$$

By Corollary 5.2, $c, d \in \mathscr{C}_{\mathrm{Par}}$ implies that c and d are seq-prime. Thus, by substitutivity and Theorem 5.1,

$$c; p_1; \ldots; p_n \sim_t d; q_1; \ldots; q_m$$

are identical factorizations into seq-primes. This implies that $c \sim_t d$ and, for each $1 \leqslant i \leqslant n = m$, $p_i \sim_t q_i$. By substitutivity, we then get that $p \sim_t q$. ∎

The dual statement of Theorem 5.2, the parallel decomposition theorem, may now be shown by following a similar strategy. First of all, we prove a unique parallel factorization result for states, which states that each $s \in \mathscr{S}$ may be expressed uniquely, modulo $\sim_t$, as a parallel composition of "semantically sequential states." Intuitively, a state is "semantically sequential" if it cannot be expressed, modulo $\sim_t$, as a nontrivial parallel composition of states. Formally:

DEFINITION 5.4 (Par-irreducible and Par-prime States). Let $s \in \mathscr{S}$. Then s is said to be:

(i) *par-irreducible* if $s \sim_t s_1 \mid s_2$ implies $s_1 \sim_t nil$ or $s_2 \sim_t nil$;

(ii) *par-prime* if s is par-irreducible and $s \not\sim_t nil$.

The proof of the unique parallel factorization result for states modulo $\sim_t$ uses the following simplification lemma, which is familiar from the theory of several equivalences based on the notion of bisimulation; see, e.g., [CH89, Mol89].

LEMMA 5.8 (Simplification Lemma). *Let $c, d, f \in \mathscr{S}$. Then:*

(1) *For each $e \in \mathbf{A}_s$, $f' \in \mathscr{S}$, $f \stackrel{e}{\Longrightarrow} f'$ and $c \mid f \sim_t d \mid f'$ imply that $d \stackrel{e}{\Longrightarrow} d'$, for some d' such that $c \sim_t d'$.*

(2) *$c \mid f \sim_t d \mid f$ implies that $c \sim_t d$.*

Proof. Both statements are proved simultaneously by induction on the combined size of c, d, and f. We assume, as an inductive hypothesis, that (1) and (2) hold for each c', d' and f' such that $|c'| + |d'| + |f'| < |c| + |d| + |f|$.

(1) Assume that $f \stackrel{e}{\Longrightarrow} f'$ and $c \mid f \sim_t d \mid f'$. Then, by the operational semantics, $c \mid f \stackrel{e}{\Longrightarrow} c \mid f'$. As $c \mid f \sim_t d \mid f'$ then is a matching e-move from $d \mid f'$. We distinguish two possibilities:

- $d \mid f' \stackrel{e}{\Longrightarrow} d' \mid f' \sim_t c \mid f'$ because $d \stackrel{e}{\Longrightarrow} d'$. Then we may apply the inductive hypothesis for (2) to obtain $c \sim_t d'$. Hence there exists d' such that $d \stackrel{e}{\Longrightarrow} d' \sim_t c$.
- $d \mid f' \stackrel{e}{\Longrightarrow} f \mid f'' \sim_t c \mid f'$ because $f' \stackrel{e}{\Longrightarrow} f''$. Then we may apply the inductive hypothesis for (1) to obtain that $d \stackrel{e}{\Longrightarrow} d' \sim_t c$, for some d'.

This completes the inductive step for statement (1).

(2) Assume $c \mid f \sim_t d \mid f$ and $c \stackrel{e}{\Longrightarrow} c'$. Then, by the operational semantics, $c \mid f \stackrel{e}{\Longrightarrow} c' \mid f$. Let us examine the possible matching moves from $d \mid f$:

- $d \mid f \stackrel{e}{\Longrightarrow} d' \mid f \sim_t c' \mid f$ because $d \stackrel{e}{\Longrightarrow} d'$. Then, by the inductive hypothesis for (2), $c' \sim_t d'$.
- $d \mid f \stackrel{e}{\Longrightarrow} d \mid f' \sim_t c' \mid f$ because $f \stackrel{e}{\Longrightarrow} f'$. Then, by the inductive hypothesis for (1), $d \stackrel{e}{\Longrightarrow} d' \sim_t c'$ for some d'.

Thus for every move of c we may find a matching move by d. By symmetry, $c \sim_t d$. ∎

For each $s \in \mathcal{S}$, a parallel factorization for s modulo $\sim_t$ is given by a set $\{c_1, ..., c_k\}$, $k \geqslant 0$, of par-primes such that

$$s \sim_t c_1 \mid \cdots \mid c_k.$$

We now have all the technical material which is needed to prove that each state has a *unique* parallel factorization into par-primes modulo $\sim_t$. The proof of the parallel factorization result follows the one of Theorem 4.2.2 (pp. 77–79) of [Mol89].

THEOREM 5.3 (Unique Parallel Factorization). *Each $s \in \mathcal{S}$ may be expressed uniquely, modulo $\sim_t$, as a parallel composition of par-primes.*

Proof. The proof is divided into two parts, an existence part and a uniqueness part, both of which are shown by induction on $|s|$, the size of $s \in \mathcal{S}$.

Existence. We show, first of all, that s can be expressed, up to $\sim_t$, as a parallel composition of par-primes. We assume, as inductive hypothesis, that the claim holds for all s' such that $|s'| < |s|$. The proof proceeds by an analysis on the possible forms s may take, modulo $\sim_t$.

It $s \sim_t nil$ then s can be expressed as an empty product of par-primes. If s is par-prime then it is its own parallel factorization. Otherwise, we may assume that $s \sim_t s_1 \mid s_2$ for some s_1, s_2 such that $|s_1|, |s_2| > 0$. By the inductive hypothesis, s_1 and s_2 have prime factorizations given by $c_1 \mid \cdots \mid c_n$ and $d_1 \mid \cdots \mid d_m$, repectively. By substitutivity, we then have that

$$s \sim_t s_1 \mid s_2 \sim_t c_1 \mid \cdots \mid c_n \mid d_1 \mid \cdots \mid d_m$$

is a parallel factorization for s. This completes the proof of the existence part of the theorem.

Uniqueness. We assume, as inductive hypothesis, that each s' such that $|s'| < |s|$ has a unique parallel factorization up to $\sim_t$. Assume now that

$$s \sim_t c_1 \mid \cdots \mid c_n \sim_t d_1 \mid \cdots \mid d_m$$

are two parallel factorizations of s into par-primes. We prove that the two factorizations are identical, modulo $\sim_t$. We proceed by analyzing the following two possibilities:

(a) the two factorizations have a common par-prime factor,

(b) $c_i \not\sim_t d_j$, for all i, j, i.e., there is no common par-prime factor in the two factorizations.

We examine the two possibilities separately.

(a) Assume, wlog, that $c_1 \sim_t d_1$. Then, by substitutivity,

$$s \sim_t c_1 | \cdots | c_n \sim_t c_1 | d_2 | \cdots | d_m.$$

By the Simplification Lemma, this implies that $c_2 | \cdots | c_n \sim_t d_2 | \cdots | d_m$. By the inductive hypothesis, which is applicable as $|c_1| > 0$, we have that $\{c_2, ..., c_n\}$ and $\{d_2, ..., d_m\}$ are identical par-prime factorizations, modulo $\sim_t$. Hence $\{c_1, ..., c_n\}$ and $\{d_1, ..., d_m\}$ are identical par-prime factorizations for s.

(b) Assume that $c_i \not\sim_t d_j$, for all i, j. If $n = 0$ then it must be the case that $m = 0$ and the two par-prime factorizations are both empty. Assume now, for the sake of contradiction, that $n \geqslant 1$. We distinguish two possibilities depending on whether $n = 1$ or $n > 1$.

(b.1) Assume that $n = 1$. Then, as c_1 is par-prime, we have that $m = 1$ and $c_1 \sim_t d_1$. This contradicts the assumption that the two decompositions have no common par-prime factor.

(b.2) By symmetry, we may assume that $n, m \geqslant 2$. Let, wlog, c_1 be minimal with respect to size amongst all the par-prime factors c_i, $1 \leqslant i \leqslant n$, and d_j, $1 \leqslant j \leqslant m$; i.e.,

$$|c_1| \leqslant |c_i| \text{ and } |c_1| \leqslant |d_j|, \qquad \text{for all } i, j.$$

Then, as $|c_1| > 0$, $c_1 \stackrel{e}{\Longrightarrow} c_1'$ for some $e \in \mathbf{A}_s$ and $c_1' \in \mathscr{S}$. By the inductive hypothesis, c_1' has a unique parallel factorization

$$c_1' \sim_t c^1 | \cdots | c^h, \qquad h \geqslant 0.$$

By the operational semantics, $c_1 | \cdots | c_n \stackrel{e}{\Longrightarrow} c_1' | c_2 | \cdots | c_n \equiv \bar{c}$ and, by the inductive hypothesis, $\bar{c}$ has a unique par-prime factorization given by

$$c^1 | \cdots | c^h | c_2 | \cdots | c_n.$$

As $c_1 | \cdots | c_n \sim_t d_1 | \cdots | d_m$, we have that, wlog, $d_1 | \cdots | d_m \stackrel{e}{\Longrightarrow} d_1' | d_2 | \cdots | d_m \sim_t \bar{c}$ because $d_1 \stackrel{e}{\Longrightarrow} d_1'$. By the inductive hypothesis, d_1' has a unique

par-prime factorization given by $d^1 \mid \cdots \mid d^k$, $k \geqslant 0$. Then, again by the inductive hypothesis,

$$c^1 \mid \cdots \mid c^h \mid c_2 \mid \cdots \mid c_n \sim_t d^1 \mid \cdots \mid d^k \mid d_2 \mid \cdots \mid d_m$$

are identical par-prime factorizations for $\bar{c}$. Note that, for each $1 \leqslant l \leqslant h$ and $2 \leqslant j \leqslant m$,

$$|c^l| < |c_1| \leqslant |d_j|.$$

Thus, for such l, j, we have that $c^l \not\sim_t d_j$. This implies that each d_j does not appear in $c^1 \mid \cdots \mid c^h \mid c_2 \mid \cdots \mid c_n$. Hence $m < 2$. This contradicts the assumption that $m \geqslant 2$. This completes the proof of the existence part.

The proof of the theorem is now complete. ▮

We shall now show how the unique parallel factorization result may be used to give a proof of the parallel decomposition theorem. As it may be expected, given the dual roles played by sequential and parallel composition and configurations in our technical development, the key to the proof of the parallel decomposition theorem is to show that each sequential configuration is par-prime.

PROPOSITION 5.5. *Each sequential configuration is par-prime.*

Proof. Let $c \in \mathscr{C}_{\mathrm{Seq}}$. We show that c is par-prime by a case analysis on the possible form c may take.

• $c \equiv F(a); p$. Assume, for the sake of contradiction, that $c \sim_t c_1 \mid c_2$ with $|c_1|, |c_2| > 0$. We proceed by analyzing the possible form of c_1 and c_2. If $c_1 \in \mathbf{P}$ then, as $|c_1| > 0$, we have that $c_1 \overset{S(a)}{\Longrightarrow}$, for some $a \in \mathbf{A}$. This implies that $c_1 \mid c_2 \overset{S(a)}{\Longrightarrow}$, contradicting the assumption that $c \sim_t c_1 \mid c_2$.

Otherwise, by symmetry, we may assume that $c_1, c_2 \in \mathscr{S} - \mathbf{P}$. Then we have that $c_1 \overset{F(a)}{\Longrightarrow}$ and $c_2 \overset{F(b)}{\Longrightarrow}$, for some a, b. This implies that $c_1 \mid c_2 \overset{F(a)}{\Longrightarrow} \overset{F(b)}{\Longrightarrow}$, again contradicting the assumption that $c \sim_t c_1 \mid c_2$.

Thus each sequential configuration of the form $F(a); p$ is par-irreducible and obviously par-prime.

• $c \equiv d; p$ with $d \in \mathscr{C}_{\mathrm{Par}}$ and $|p| > 0$. Assume, for the sake of contradiction, that $c \sim_t c_1 \mid c_2$, with $|c_1|, |c_2| > 0$. By Lemma 5.3(ii), this implies that $c_1 \mid c_2 \in \mathbf{P}$ or $c_1 \mid c_2 \in \mathscr{C}_{\mathrm{Par}}$. If $c_1 \mid c_2 \in \mathbf{P}$ then we obtain a contradiction to $c \sim_t c_1 \mid c_2$ as $c \overset{F(a)}{\Longrightarrow}$ for some a.

If $c_1 \mid c_2 \in \mathscr{C}_{\mathrm{Par}}$ then, by the sequential decomposition theorem, $c \equiv d; p \sim_t c_1 \mid c_2$ implies $p \sim_t nil$. This contradicts the assumption that $|p| > 0$. Thus each sequential configuration of the form $d; p$ is par-prime.

As we have considered all the possible forms sequential configurations may take, we have that each $c \in \mathscr{C}_{\mathrm{Seq}}$ is par-prime. ∎

We can now prove the promised parallel decomposition theorem.[1] In the statement of this result we use some new notation which is introduced in the following definition.

DEFINITION 5.5. Let $\{c_i \mid i \in I\}$, $\{d_j \mid j \in J\} \subseteq \mathscr{C}_{\mathrm{Seq}}$, with I and J finite index sets. Then we write $\{c_i \mid i \in I\} \sim_{\mathrm{t}} \{d_j \mid j \in J\}$ iff there exists a bijective map $\phi: I \to J$ such that $c_i \sim_{\mathrm{t}} d_{\phi(i)}$, for each $i \in I$.

THEOREM 5.4 (Parallel Decomposition Theorem). *Let* $\{c_i \mid i \in I\}$, $\{d_j \mid j \in J\} \subseteq \mathscr{C}_{\mathrm{Seq}}$, $p, q \in \mathbf{P}$. *Then* $c \equiv (\prod_{i \in I} c_i) \mid p \sim_{\mathrm{t}} (\prod_{j \in J} d_j) \mid q \equiv d$ *implies* $\{c_i \mid i \in I\} \sim_{\mathrm{t}} \{d_j \mid j \in J\}$ *and* $p \sim_{\mathrm{t}} q$.

Proof. By the unique parallel factorization theorem (Theorem 5.3), there exist unique par-prime factorizations for p and q given by

$$\text{(i)} \quad p \sim_{\mathrm{t}} p_1 \mid \cdots \mid p_n,$$

$$\text{(ii)} \quad q \sim_{\mathrm{t}} q_1 \mid \cdots \mid q_m.$$

Note that each par-prime factor of p and q has to be a process. By Proposition 5.5, each c_i and d_j is par-prime as it is a sequential configuration. Thus, by substitutivity and the unique parallel factorization theorem,

$$\left(\prod_{i \in I} c_i\right) \mid p_1 \mid \cdots \mid p_n \sim_{\mathrm{t}} \left(\prod_{j \in J} d_j\right) \mid q_1 \mid \cdots \mid q_m$$

are identical par-prime factorizations up to $\sim_{\mathrm{t}}$. As, obviously, $c_i \not\sim_{\mathrm{t}} q_k$ and $d_j \not\sim_{\mathrm{t}} p_h$, for all i, j, h, k, it must be the case that

$$\{c_i \mid i \in I\} \sim_{\mathrm{t}} \{d_j \mid j \in J\} \qquad \text{and} \qquad \{p_h \mid 1 \leqslant h \leqslant n\} \sim_{\mathrm{t}} \{q_k \mid 1 \leqslant k \leqslant m\}.$$

By substitutivity, (i), and (ii), we then have that $p \sim_{\mathrm{t}} q$. This completes the proof of the theorem. ∎

Intuitively, for a process $p \in \mathbf{P}$, its unique parallel factorization modulo $\sim_{\mathrm{t}}$ may be seen as the "most parallel version" of p (modulo $\sim_{\mathrm{t}}$). Similarly, its unique sequential factorization may be seen as the "most sequential version" of p (modulo $\sim_{\mathrm{t}}$). For each $p \in \mathbf{P}$, by the two factorization theorems we have just shown,

$$\begin{aligned} p &\sim_{\mathrm{t}} p_1 \mid \cdots \mid p_n && \text{a unique product of par-primes} \\ &\sim_{\mathrm{t}} q_1 ; \ldots ; q_m && \text{a unique sequential composition of seq-primes.} \end{aligned}$$

[1] We thank Frits Vaandrager for suggesting the proof of this theorem which is presented below.

With respect to standard bisimulation, $\sim$, it is possible to have both the factorizations be nontrivial, which we take to mean that $n, m \geqslant 2$. For instance, for $p \equiv a \mid a$,

$$p \sim_t a \mid a \qquad \text{(parallel factorization)}$$

$$\sim_t a; a \qquad \text{(sequential factorization)}.$$

We now show that, up to $\sim_t$, at least one of the factorizations is trivial.

THEOREM 5.5. *Let $p \in \mathbf{P}$. Assume that $p_1 \mid \cdots \mid p_n$ and $q_1; \ldots; q_m$ are a par-prime factorization and a seq-prime factorization for p, up to $\sim_t$, respectively. Then $n = m = 0$ or $n = 1$ or $m = 1$.*

Proof. Assume, for the sake of contradiction, that $n, m \geqslant 2$ and that

$$p_1 \mid \cdots \mid p_n \sim_t q_1; \ldots; q_m. \tag{4}$$

As $|p_1| > 0$, there exist $a \in \mathbf{A}$ and $c \in \mathscr{S}$ such that $p_1 \xrightarrow{S(a)} c$. By the operational semantics, we then have that $p_1 \mid \cdots \mid p_n \xrightarrow{S(a)} c \mid \bar{p}$, where $\bar{p} \equiv p_2 \mid \cdots \mid p_n$ and $|\bar{p}| > 0$ as $n \geqslant 2$. By (4) and the fact that $|q_1| > 0$, we have that, for some d, $q_1; \ldots; q_m \xrightarrow{S(a)} d; \bar{q} \sim_t c \mid \bar{p}$, where $\bar{q} \equiv q_2; \ldots; q_m$ and $|\bar{q}| > 0$ as $m \geqslant 2$. By Lemma 5.4, $c, d \in \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$. Moreover, as $c \mid \bar{p} \sim_t d; \bar{q}$ and $|\bar{p}| > 0$, we have that $d \in \mathscr{C}_{\mathrm{Par}}$ or d is of the form $\vec{d}; t$ with $\vec{d} \in \mathscr{C}_{\mathrm{Par}}$ and $|t| > 0$. In both cases $d; \bar{q} \in \mathscr{C}_{\mathrm{Seq}}$.

We proceed by considering the cases $c \in \mathscr{C}_{\mathrm{Seq}}$ and $c \in \mathscr{C}_{\mathrm{Par}}$ separately. If $c \in \mathscr{C}_{\mathrm{Seq}}$ then, by the parallel decomposition theorem, $c \mid \bar{p} \sim_t d; \bar{q}$ implies that $\bar{p} \sim_t nil$. This contradicts the fact that $|\bar{p}| > 0$.

If $c \in \mathscr{C}_{\mathrm{Par}}$ then, by Lemma 5.4, c has the form $c_1 \mid t$ for some $c_1 \in \mathscr{C}_{\mathrm{Seq}}$ and t such that $|t| > 0$. Then, by substitutivity and the parallel decomposition theorem, $c \mid \bar{p} \sim_t c_1 \mid t \mid \bar{p} \sim_t d; \bar{q}$ implies that $t \mid \bar{p} \sim_t nil$. This contradicts the hypothesis that $|t|, |\bar{p}| > 0$.

Thus we have shown that either $n < 2$ or $m < 2$. Obviously, if $n = 0$ then $m = 0$. Hence, by symmetry, $n = m = 0$ or $n = 1$ or $m = 1$. ∎

An interesting corollary of the above theorem is the following result stating that, up to $\sim_t$, each process is either *nil* or par-prime or seq-prime.

COROLLARY 5.3. *Let $p \in \mathbf{P}$. Then $p \sim_t nil$ or p is par-prime or p is seq-prime.*

5.3. *The Completeness Theorem*

In order to prove the promised completeness result, we will need some further lemmas whose proofs will highlight the usefulness of the important decomposition results shown in the previous section.

LEMMA 5.9. *Let* $c, d \in \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$. *Then*:

(i) $c \in \mathscr{C}_{\mathrm{Seq}}$ *and* $c \sim_{\mathrm{t}} d$ *imply* $d \in \mathscr{C}_{\mathrm{Seq}}$.

(ii) $c \in \mathscr{C}_{\mathrm{Par}}$ *and* $c \sim_{\mathrm{t}} d$ *imply* $d \in \mathscr{C}_{\mathrm{Par}}$.

Proof. We just prove (i) as (ii) then follows by symmetry. Assume that $c \in \mathscr{C}_{\mathrm{Seq}}$ and $c \sim_{\mathrm{t}} d$. Suppose, towards a contradiction, that $d \in \mathscr{C}_{\mathrm{Par}}$. Then, by the definition of $\mathscr{C}_{\mathrm{Par}}$, either $d \equiv \prod_{j \in J} d_j \mid p$ with $|J| > 0$ and $d_j \in \mathscr{C}_{\mathrm{Seq}}$ for each $j \in J$ and $|p| > 0$, or $d \equiv \prod_{j \in J} d_j$ with $|J| > 1$ and $d_j \in \mathscr{C}_{\mathrm{Seq}}$ for each $j \in J$. If $d \equiv \prod_{j \in J} d_j \mid p \sim_{\mathrm{t}} c$ then, by Theorem 5.4, we obtain that $p \sim_{\mathrm{t}} nil$. This contradicts the assumption that $|p| > 0$. If $d \equiv \prod_{j \in J} d_j \sim_{\mathrm{t}} c$ then, again by Theorem 5.4, we obtain that $\{c\} \sim_{\mathrm{t}} \{d_j \mid j \in J\}$. However, this is impossible as $|J| > 1$. Hence $d \in \mathscr{C}_{\mathrm{Seq}}$. ∎

LEMMA 5.10. *Let* $c, d \in \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$. *Then*:

(1) $c \equiv F(a); p \sim_{\mathrm{t}} d$ *implies* $d \equiv F(a); q$ *and* $p \sim_{\mathrm{t}} q$.

(2) $c \equiv c_1; p \sim_{\mathrm{t}} d$, *with* $c_1 \in \mathscr{C}_{\mathrm{Par}}$ *and* $|p| > 0$, *implies* $d \equiv d_1; q$ *with* $d_1 \in \mathscr{C}_{\mathrm{Par}}$, $|q| > 0$, $c_1 \sim_{\mathrm{t}} d_1$, *and* $p \sim_{\mathrm{t}} q$.

(3) $c \equiv c_1 \mid p \sim_{\mathrm{t}} d$, *with* $c_1 \in \mathscr{C}_{\mathrm{Seq}}$ *and* $|p| > 0$, *implies* $d \equiv d_1 \mid q$, *with* $d_1 \in \mathscr{C}_{\mathrm{Seq}}$, $|q| > 0$, $c_1 \sim_{\mathrm{t}} d_1$, *and* $p \sim_{\mathrm{t}} q$.

Proof. Assume $c, d \in \mathscr{C}_{\mathrm{Seq}} \cup \mathscr{C}_{\mathrm{Par}}$. We prove each statement separately.

(1) Assume $c \equiv F(a); p \sim_{\mathrm{t}} d$. Then $c \in \mathscr{C}_{\mathrm{Seq}}$ and, by Lemma 5.9, $c \sim_{\mathrm{t}} d$ implies $d \in \mathscr{C}_{\mathrm{Seq}}$ as well. We show, first of all, that d must be of the form $F(a); q$. Assume in fact, towards a contradiction, that d is of the form $d_1; q$, with $d_1 \in \mathscr{C}_{\mathrm{Par}}$ and $|q| > 0$. Then, as $d_1 \in \mathscr{C}_{\mathrm{Par}}$, either $d_1 \xRightarrow{S(b)}$ or $d_1 \xRightarrow{F(b)} \xRightarrow{F(b')}$, for some $b, b' \in \mathbf{A}$. By the operational semantics the same is true of d, which contradicts the hypothesis that $c \sim_{\mathrm{t}} d$. Hence d must be of the form $F(a); q$ and this easily implies $p \sim_{\mathrm{t}} q$.

(2) Assume $c \equiv c_1; p \sim_{\mathrm{t}} d$, with $c_1 \in \mathscr{C}_{\mathrm{Par}}$ and $|p| > 0$. Then $c \in \mathscr{C}_{\mathrm{Seq}}$ and, by Lemma 5.9, $c \sim_{\mathrm{t}} d$ implies $d \in \mathscr{C}_{\mathrm{Seq}}$ as well. By (1) and symmetry, we have that d must be of the form $d_1; q$, with $d_1 \in \mathscr{C}_{\mathrm{Par}}$ and $|q| > 0$. Thus, $c \equiv c_1; p \sim_{\mathrm{t}} d_1; q \equiv d$ and we may apply Corollary 5.2 to obtain $c_1 \sim_{\mathrm{t}} d_1$ and $p \sim_{\mathrm{t}} q$.

(3) Assume $c \equiv c_1 \mid p \sim_{\mathrm{t}} d$, with $c_1 \in \mathscr{C}_{\mathrm{Seq}}$ and $|p| > 0$. Then $c \in \mathscr{C}_{\mathrm{Par}}$ and, by Lemma 5.9, $c \sim_{\mathrm{t}} d$ implies $d \in \mathscr{C}_{\mathrm{Par}}$ as well. We show, first of all, that d must be of the form $d_1 \mid q$, with $d_1 \in \mathscr{C}_{\mathrm{Seq}}$ and $|q| > 0$. Assume in fact, towards a contradiction, that $d \equiv \prod_{j \in J} d_j$, with $|J| > 1$ and $d_j \in \mathscr{C}_{\mathrm{Seq}}$ for each $j \in J$. Then we may apply Theorem 5.4 to $c_1 \mid p \sim_{\mathrm{t}} \prod_{j \in J} d_j$ to obtain $p \sim_{\mathrm{t}} nil$. This contradicts the hypothesis that $|p| > 0$.

Assume now $d \equiv \prod_{j \in J} d_j \mid q$, with $|J| > 0$, $|q| > 0$ and $d_j \in \mathscr{C}_{\mathrm{Seq}}$ for each $j \in J$. Then we may apply Theorem 5.4 to $c_1 \mid p \sim_{\mathrm{t}} \prod_{j \in J} d_j \mid q$ to obtain

$\{c_1\} \sim_t \{d_j \mid j \in J\}$ and $p \sim_1 q$. Hence J is a singleton set and the statement holds. ∎

The following result, which plays an important role in the proof of the completeness theorem, expresses a very strong property of $\sim_t$ when applied to sequential and parallel head reduced forms.

PROPOSITION 5.6. *Let* $p, q \in \mathrm{HRF}_{\mathrm{Seq}} \cup \mathrm{HRF}_{\mathrm{Par}}$. *Assume* $p \overset{S(a)}{\Longrightarrow} c$, $q \overset{S(a)}{\Longrightarrow} d$ *and* $c \sim_t d$. *Then* $p \sim_t q$.

Proof. By induction on the combined size of p and q. We assume, as inductive hypothesis, that the claim holds for all $p', q' \in \mathrm{HRF}_{\mathrm{Seq}} \cup \mathrm{HRF}_{\mathrm{Par}}$ such that $|p'| + |q'| < |p| + |q|$. Assume that $p \overset{S(a)}{\Longrightarrow} c$, $q \overset{S(a)}{\Longrightarrow} d$, and $c \sim_t d$. By Lemma 5.3, $c \in \mathscr{C}_{\mathrm{Seq}}$ or c is of the form $c' \mid r$, with $c' \in \mathscr{C}_{\mathrm{Seq}}$ and $|r| > 0$. We consider these two possibilities in turn.

- $c \in \mathscr{C}_{\mathrm{Seq}}$. We distinguish two subcases according to the structure of c.

1. $c \equiv F(a); r$. Then, by Lemma 5.10, $c \sim_t d$ implies that $d \equiv F(a); t$ and $r \sim_t t$. It is now easy to see that $p \overset{S(a)}{\Longrightarrow} F(a); r$, $p \in \mathrm{HRF}_{\mathrm{Seq}} \cup \mathrm{HRF}_{\mathrm{Par}}$, implies that p is of the form $a; r$. Similarly q is of the form $a; t$. As $r \sim_t t$ we then have, by substitutivity, that $p \sim_t q$.

2. $c \equiv c'; r$, with $c' \in \mathscr{C}_{\mathrm{Par}}$ and $|r| > 0$. Then, by Lemma 5.10, $c \sim_t d$ implies that d is of the form $d'; t$, with $d' \in \mathscr{C}_{\mathrm{Par}}$, $|t| > 0$, $c' \sim_t d'$ and $r \sim_t t$. As $p \overset{S(a)}{\Longrightarrow} c$ and $p \in \mathrm{HRF}_{\mathrm{Seq}} \cup \mathrm{HRF}_{\mathrm{Par}}$, p must be of the form $p_1; r$, with $p_1 \in \mathrm{HRF}_{\mathrm{Par}}$ and $p_1 \overset{S(a)}{\Longrightarrow} c'$. Similarly q has to be of the form $q_1; t$, with $q_1 \in \mathrm{HRF}_{\mathrm{Par}}$ and $q_1 \overset{S(a)}{\Longrightarrow} d'$. As $|r|, |t| > 0$, $|q_1| + |p_1| < |p| + |q|$ and we may apply induction to obtain $p_1 \sim_t q_1$. By substitutivity, $p \sim_t q$.

- $c \equiv c' \mid r$, with $c' \in \mathscr{C}_{\mathrm{Seq}}$ and $|r| > 0$. By Lemma 5.10, $c \sim_t d$ implies d is of the form $d' \mid t$, with $d' \in \mathscr{C}_{\mathrm{Seq}}$, $|t| > 0$, $c' \sim_t d'$, and $r \sim_t t$. It is easy to see that $p \overset{S(a)}{\Longrightarrow} c' \mid r$ and $p \in \mathrm{HRF}_{\mathrm{Seq}} \cup \mathrm{HRF}_{\mathrm{Par}}$ imply $p \equiv p_1 \mid r$, with $p_1 \in \mathrm{HRF}_{\mathrm{Seq}}$ and $p_1 \overset{S(a)}{\Longrightarrow} c'$. Similarly, $q \overset{S(a)}{\Longrightarrow} d' \mid t$ implies $q \equiv q_1 \mid t$, with $q_1 \in \mathrm{HRF}_{\mathrm{Seq}}$ and $q_1 \overset{S(a)}{\Longrightarrow} d'$. As $|r|, |t| > 0$, we may apply the inductive hypothesis to obtain $p_1 \sim_t q_1$. Hence, by substitutivity, $p \sim_t q$. ∎

We have now developed all the technical machinery needed in the proof of the completeness theorem.

THEOREM 5.6 (Completeness). *Let* $p, q \in \mathbf{P}$. *Then* $p \sim_t q$ *implies* $p =_E q$.

Proof. The proof is by induction on the combined size of the terms. By the reduction lemma and the soundness of $=_E$ we may assume, wlog, that $p \equiv \sum_{i \in I} p_i$ and $q \equiv \sum_{j \in J} q_j$ are head reduced forms. By symmetry, it is then sufficient to show that

$$\forall i \in I \ \exists j \in J: p_i =_E q_j.$$

In fact, the equality $p =_E q$ will then be provable by applications of (A1)–(A3). Let $i \in I$. The proof proceeds by an analysis of the possible structure of p_i.

• $p_i \equiv a; r, r \in \mathrm{HRF}$. Then $p_i \xrightarrow{S(a)} F(a); r$. By the operational semantics, $p_i \xrightarrow{S(a)} F(a); r$ implies $p \xrightarrow{S(a)} F(a); r$. As $p \sim_t q$, there exists q_j such that $q_j \xrightarrow{S(a)} c \sim_t F(a); r$. By Lemma 5.10, $c \sim_t F(a); r$ implies $c \equiv F(a); t$ with $r \sim_t t$. By the inductive hypothesis, $r =_E t$. Moreover, $q_j \xrightarrow{S(a)} F(a); t$ and $q_j \in \mathrm{HRF}_{\mathrm{Seq}} \cup \mathrm{HRF}_{\mathrm{Par}}$ imply $q_j \equiv a; t$. Hence, by substitutivity, $p_i \equiv a; r =_E a; t \equiv q_j$.

• $p_i \equiv r_1; r_2$, with $r_1 \in \mathrm{HRF}_{\mathrm{Par}}$, $r_2 \in \mathrm{HRF}$ and $|r_2| > 0$. Assume $p_i \xrightarrow{S(a)} c$. Then c must be of the form $c'; r_2$ with $r_1 \xrightarrow{S(a)} c'$. By Lemma 5.3, $c' \in \mathscr{C}_{\mathrm{Par}}$ and thus $c \in \mathscr{C}_{\mathrm{Seq}}$. As $p \sim_t q$, there exists q_j such that $q_j \xrightarrow{S(a)} d \sim_t c'; r_2$. By Lemma 5.10, $d \sim_t c'; r_2$ implies d has the form $d'; t_2$, with $d' \in \mathscr{C}_{\mathrm{Par}}$, $|t_2| > 0$, $c' \sim_t d'$ and $r_2 \sim_t t_2$. By the inductive hypothesis, $r_2 =_E t_2$.

It is now easy to see that $q_j \xrightarrow{S(a)} d'; t_2$ and $q_j \in \mathrm{HRF}_{\mathrm{Seq}} \cup \mathrm{HRF}_{\mathrm{Par}}$ imply $q_j \equiv t_1; t_2$, with $t_1 \in \mathrm{HRF}_{\mathrm{Par}}$ and $t_1 \xrightarrow{S(a)} d'$. By Proposition 5.6, $r_1 \xrightarrow{S(a)} c'$, $t_1 \xrightarrow{S(a)} d'$ and $c' \sim_t d'$ imply $r_1 \sim_t t_1$. Hence, by the inductive hypothesis, $r_1 =_E t_1$ and, by substitutivity, $p_i \equiv r_1; r_2 =_E t_1; t_2 \equiv q_j$.

• $p_i \equiv r_1 \between r_2$, with $r_1 \in \mathrm{HRF}_{\mathrm{Seq}}$, $r_2 \in \mathrm{HRF}$ and $|r_2| > 0$. Assume $p_i \xrightarrow{S(a)} c$. Then c must be of the form $c_1 \mid r_2$, with $r_1 \xrightarrow{S(a)} c_1$. By Lemma 5.3, $c_1 \in \mathscr{C}_{\mathrm{Seq}}$. As $p \sim_t q$, there exists q_j such that $q_j \xrightarrow{S(a)} d \sim_t c_1 \mid r_2$. By Lemma 5.10, it must be the case that $d \equiv d_1 \mid t_2$, with $d_1 \in \mathscr{C}_{\mathrm{Seq}}$, $|t_2| > 0$, $c_1 \sim_t d_1$ and $r_2 \sim_t t_2$. By the inductive hypothesis, $r_2 =_E t_2$.

As $q_j \xrightarrow{S(a)} d_1 \mid t_2$ and $q_j \in \mathrm{HRF}_{\mathrm{Seq}} \cup \mathrm{HRF}_{\mathrm{Par}}$, q_j must be of the form $t_1 \between t_2$, with $t_1 \in \mathrm{HRF}_{\mathrm{Seq}}$ and $t_1 \xrightarrow{S(a)} d_1$. As $c_1 \sim_t d_1$, we then have, by Proposition 5.6, that $r_1 \sim_t t_1$. By the inductive hypothesis, $r_1 =_E t_1$. By substitutivity, $p_i \equiv r_1 \between r_2 =_E t_1 \between t_2 \equiv q_j$. This completes the proof of the theorem. ∎

6. Conclusions

In this paper, we have studied the consequences of the introduction of an operator for refining an action by a process in a simple process algebra.

More specifically we have considered a process algebra which constitutes the core of many of the existing ones, added to it a new combinator for refining an action by a process, and then addressed the question of an appropriate equivalence for the augmented language.

The main result of this paper is that, at least for the simple language we have considered, an adequate equivalence relation can be defined in a very intuitive manner. In fact, it coincides with both Hennessy's timed-

equivalence [H88] and a slight reformulation of it which we call refine-equivalence. Moreover, these equivalences can be axiomatized in much the same way as the standard behavioural equivalences [HM85, DH84].

The technical machinery and the understanding of the problem that we have gained in the development of the work reported in this paper have proved to be of considerable help in its extensions to more complex algebras [AH90]. As an example produced by R. Van Glabbeek shows, timed equivalence is *not* in general preserved by action refinement over, e.g., event structures. The characterization theorem we have proved in this paper is thus language dependent and does not hold for process algebras which include a parallel operator with synchronization and **CCS** restriction. However, as we show in [AH90], many of the techniques developed in this paper for refine-equivalence are indeed applicable in general and a characterization result in terms of refine-equivalence similar to Corollary 4.3 still holds. The interested reader is invited to consult [AH90] for more details.

After having developed a more comprehensive theory of action-refinement in process algebras, we aim to carry out a detailed study of the usefulness of this notion in the specification and development of reactive systems. Our feeling is that such a feature should allow a further degree of freedom in the modularization of the development of concurrent systems, in much the same way as the notion of *procedure* has been of help in describing complex sequential systems. This claim will have to be supported by working out the specification of a range of realistic systems using the algebraic techniques we plan to develop.

Finally, let us refer briefly to some related work. In [Pr86], Pratt discusses the *pomset model* of concurrent computation. One of the most interesting operators he introduces is the operator that he calls *pomset homomorphism*. Such an operator is in essence similar to the one we have introduced in this paper. In fact, it has the effect of substituting a pomset to a vertex of another pomset.

In [Gi84], the author, working with Pratt's pomset model, shows that, regarding each symbol of a pomset as standing for a language, causal dependency between symbols as concatenation of languages and concurrency as their shuffle, two pomsets are equivalent (in the sense that they stand for the same language) if and only if they are equivalent under the interpretation in which only languages with strings of length at most 2 are considered.

In [K88], A. Kiehn introduces a call mechanism for Petri net system, which are finite families of place/transition nets, [Rei85], and studies some closure properties of the class of languages accepted by them. None of these papers, however, is concerned with an explicit algebraic treatment of the

feature of action-refinement in the different models adopted by the authors.

In [BC88], G. Boudol and I. Castellani propose a calculus of concurrent processes which allows them to regard a finite computation as a single event. They investigate this problem both at the level of *execution* of a process and at the level of *operation* (the way processes operate on data). This kind of abstraction is in essence a dual notion of the refinement operator described in this paper and it would be interesting to have a process algebra containing both these features. Since the appearance of [CDP87], semantic theories for processes which support the refinement of actions by processes have been the object of extensive study in the literature. The reference [GG88] gives a good survey of the work in this area; moreover, there the authors show that history preserving bisimulation over finite prime event structures [Win87] is preserved by action-refinement (in the absence of internal, invible actions). [NEL88] presents a natural model for a process language incorporating a refinement operator which is fully abstract with respect to a trace-based notion of equivalence over processes. [DD89b] studies action-refinement over synchronization and causal trees [DD89a] and presents refinement theorems for two versions of branching bisimulation [GW89]. In [Gl90], the author studies notions of ST-bisimulation, a version of bisimulation equivalence which is very close in spirit to the refine-equivalence presented in this paper, and ST-trace equivalence over finite, prime event structures and proves that they are both preserved by refinement. A refinement theorem for a version of failures semantics [BHR84] over safe Petri nets has been presented in [Vo90]. However, apart from [NEL88], all of the above-mentioned references are concerned with the study of action-refinement at the semantic level. In this paper, on the other hand, we have tried to provide the theoretical foundations for a syntactic treatment of the feature of action-refinement in a simple algebraic setting. Extensions of our work to more complex algebras will be reported in [AH90].

APPENDIX: TABLE OF NOTATION

$\mathbf{A}$	Set of actions ($a, b, \ldots \in \mathbf{A}$)
$\mathbf{A}_s$	Set of subactions ($e, e', \ldots \in \mathbf{A}_s$)
$\mathbf{LA}$	Set of labelled actions ($a_i, b_j, \ldots \in \mathbf{LA}$)
$\mathbf{LA}_s$	Set of labelled subactions ($\lambda \in \mathbf{LA}_s$)
$\mathbf{P}$	Set of basic processes ($p, q, \ldots \in \mathbf{P}$)
$\mathbf{P}_{\mathrm{ref}}$	Process language with refinement ($p, q, \ldots \in \mathbf{P}_{\mathrm{ref}}$)
$\mathbf{P}_\rho$	$\mathbf{P}$ with refinement combinators $[\rho]$ ($p, q, \ldots \in \mathbf{P}_\rho$)

$\mathscr{S}$	Set of states over **P** ($s, s', \ldots \in \mathscr{S}$)
LP	Set of uniquely labelled processes ($\pi, \pi', \ldots \in \mathbf{LP}$)
$\mathbf{L}\mathscr{S}$	Set of uniquely labelled states ($c, d, \ldots \in \mathbf{L}\mathscr{S}$)
$\mathscr{C}_{\mathrm{Seq}}$	Set of sequential configurations
$\mathscr{C}_{\mathrm{Par}}$	Set of parallel configurations
$\mathrm{HRF}_{\mathrm{Seq}}$	Set of sequential head reduced forms
$\mathrm{HRF}_{\mathrm{Par}}$	Set of parallel head reduced forms
HRF	Set of head reduced forms
$\xrightarrow{a}$	Transition relations over **P**
$\xrightarrow{a}_{\rho}$	Transition relations over $\mathbf{P}_{\rho}$
$\stackrel{e}{\Longrightarrow}$	Timed transition relations over $\mathscr{S}$
$\xrightarrow{c}_{i}$	Transition relations over $\mathbf{T}_{\Sigma}$
$\overset{\lambda}{\longmapsto}$	Labelled transition relations over $\mathbf{L}\mathscr{S}$
$\sim$	Bisimulation equivalence
$\sim^{c}$	Largest $\mathbf{P}_{\mathrm{ref}}$-congruence contained in $\sim$
$\sim_{t}$	Timed-equivalence
$\sim_{r}$	Refine-equivalence
$\{\sim_{\phi} \mid \phi \in \mathscr{H}\}$	Largest parameterized bisimulation
$\in \sim_{t}$	Set membership modulo $\sim_{t}$
$\mathscr{H}$	Set of histories
SUB	Set of substitutions
red$(\cdot)$	Reduction function
Tr$(\cdot)$	Translation function
Places(a, c)	Set of a-places of c
un$(\cdot)$	Function which forgets the labels of configurations
$\eta(\cdot)$	See Definition 4.9

ACKNOWLEDGMENTS

It is a pleasure to acknowledge Frits Vaandrager for suggesting the elegant proof of the Parallel Decomposition Theorem based on the unique factorization result and his detailed comments on a previous version of this paper. We also thank the anonymous referee for his/her many suggestions which led to improvements in the paper.

RECEIVED September 13, 1989; FINAL MANUSCRIPT RECEIVED February 27, 1991

REFERENCES

[Ab87] ABRAMSKY, S. (1987), Observation equivalence as a testing equivalence, *Theoret. Comput. Sci.* **53**, 225–241.

[AH90] ACETO, L., AND HENNESSY, M. (1990), "Adding Action Refinement to a Finite Process Algebra," Computer Science Report 6/90, University of Sussex, to appear in *Information and Computation.*

[BC88] BOUDOL, G., AND CASTELLANI, I. (1988), Concurrency and atomicity, *Theoret. Comput. Sci.* **59**, 25–84.

[BHR84] BROOKES, S. D., HOARE, C. A. R., AND ROSCOE, A. W. (1984), A theory of communicating sequential processes, *J. Assoc. Comput. Mach.* **31** (3), 560–599.

[BK84] BERGSTRA, J. A., AND KLOP, J. W. (1984), Process algebra for synchronous communication, *Inform. Control* **60**, 109–137.

[BK85] BERGSTRA, J. A., AND KLOP J. W. (1985). Algebra of communicating processes with abstraction, *Theoret. Comput. Sci.* **37** (1), 77–121.

[BK88] BERGSTRA, J. A., AND KLP, J. W. (1988), Process theory based on bisimulation semantics, *in* "Proceedings REX School 1988," pp. 50–122. Lecture Notes in Computer Science, Vol. 354, Berlin/New York.

[CDP87] CASTELLANO, L., DE MICHELIS, G., AND POMELLO, L. (1987), Concurrency vs interleaving: An instructive example, *Bull. European Assoc. Theoret. Comput. Sci.* **31**, 12–15.

[CH89] CASTELLANI, I., AND HENNESSY, M. (1989), Distributed bisimulations, *J. Assoc. Comput. Mach.* (October), 887–911.

[DD89a] DARONDEAU, P, AND DEGANO, P. (1989), Causal trees, *in* "Proceedings ICALP 1989," pp. 234–248, Lecture Notes in Computer Science, Vol. 372, Springer-Verlag, Berlin/New York.

[D89b] DARONDEAU, P., AND DEGANO, P. About semantic action refinement, *Fund. Inform.*, to appear.

[DH84] DE NICOLA, R., AND HENNESSY, M. (1984), Testing equivalences for processes, *Theoret. Comput. Sci.* **34** (1), 83–134.

[Gi84] GISCHER, J. L. (1984), "Partial Orders and the Axiomatic Theory of Shuffle," Ph.D. Thesis, Stanford University.

[GG88] VAN GLABBEEK, R., AND GOLTZ, U. (1988), "Equivalence Notions for Concurrent Systems and Refinement of Actions," Arbeitspapiere de GMD 366.

[Gl90] VAN GLABBEEK, R. (1990), The refinement theorem for *ST*-bisimulation semantics, to appear *in* "Proceedings IFIP Working Conference, Israel at the Sea of Galilee."

[Gl90a] VAN GLABBEEK, R. (1990), "Comparative Concurrency Semantics and Refinement of Actions," Ph.D. Thesis, Free University of Amsterdam.

[GW89] VAN GLABBEEK, R., AND WEIJLAND, P. (1989), Branching time and abstraction in bisimulation semantics, *in* "Information Processing 1989," pp. 613–618, Elsevier, Amsterdam/New York.

[GW89a] VAN GLABBEEK, R., AND WEIJLAND, P. (1989), Refinement in branching time semantics, *in* "Proceedings AMAST Conference, May 1989, Iowa," pp. 197–201.

[H88] HENNESSY, M. (1988), Axiomatising finite concurrent processes, *SIAM J. Comput.* (October).

[HM85] HENNESSY, M., AND MILNER, R. (1985), Algebraic laws for nondeterminism and concurrency, *J. Assoc. Comput. Mach.* **32** (1), 137–161.

[Hoare85] HOARE, C. A. R. (1985), "Communicating Sequential Processes," Prentice–Hall, Englewood Cliffs, NJ.

[K88] KIEHN, A. (1988), "Petri Net Systems and Their Closure Properties," Technische Universität München, Institut für Informatik; to appear *in* "Advances in Petri Nets," 1989.

[Kel76] KELLER, R. (1976), Formal verification of parallel programs, *Comm. ACM* **19** (7), 561–572.

[Mil80] MILNER, R. (1980), "A Calculus of Communicating Systems," Lecture Notes in Computer Science, Vol. 92, Springer-Verlag, Berlin/New York.

[Mil83] MILNER, R. (1983), Calculi for synchrony and asynchrony, *Theoret. Comput. Sci.* **25**, 267–310.

[Mil89] MILNER, R. (1989), "Communication and Concurrency," Prentice–Hall, Englewood Cliffs, NJ.

[MM90] MILNER, R., AND MOLLER, F. (1990), Unique decomposition of processes, *Bull. European Assoc. Theoret. Comput. Sci.* **41**, 226–232.

[Mol89] MOLLER, F. (1989), "Axioms for Concurrency," Ph.D. Thesis, Report CS-59-89, Department of Computer Science, University of Edinburgh.

[NEL88] NIELSEN, M., ENGBERG, U., AND LARSEN, K. S. (1988), Fully abstract models for a process language with refinement, *in* "Proceedings REX School 1988," pp. 523–548, Lecture Notes in Computer Science, Vol. 354, Springer-Verlag, Berlin/New York.

[OC84] INMOS Limited (1984), "OCCAM Programming Manual," Prentice–Hall, London.

[Par81] PARK, D. (1981), Concurrency and automata on infinite sequences, *in* "Lecture Notes in Computer Science," Vol. 104, pp. 167–183, Springer-Verlag, Berlin/New York.

[Pl81] PLOTKIN, G. (1981), "A Structural Approach to Operational Semantics," Report DAIMI FN-19, Computer Science Department, Aarhus University.

[Pn85] PNUELI, A. (1985), Linear and branching structures in the semantics and logics of reactive systems, *in* "Lecture Notes in Computer Science," Vol. 194, pp. 14–32, Springer-Verlag, Berlin/New York.

[Pr86] PRATT, V. (1986), Modelling concurrency with partial orders, *Internat. J. Parallel Programming* **15**, 33–71.

[Rei85] REISIG, W. (1985), "Petri Nets," EATCS Monographs on Theoretical Computer Science, Springer-Verlag, Berlin/New York.

[Vo90] VOGLER, W. (1990), Failures semantics based on interval semiwords is a congruence for refinement, *in* "Proceedings STACS 1990," Lecture Notes in Computer Science, Springer-Verlag, Berlin/New York.

[Win87] WINSKEL, G. (1987), Event structures, *in* "Advances in Petri Nets 1986," pp. 325–392, Lecture Notes in Computer Science, Vol. 255, Springer-Verlag, Berlin/New York.