On the Operational Semantics of Nondeterminism and Divergence
Erdogmus, Hakan; Johnston, R.; Ferguson, M.

Publisher's version / Version de l'éditeur:

Semantics and Theory of Programming, 160, pp. 271-317, 1996

On the Operational Semantics of Nondeterminism and Divergence *

Erdogmus, H., Johnston, R., Ferguson, M.
January 1996

* published in Theoretical Computer Science B: Logics, Semantics and Theory of Programming, 160: 271-317. January 1996. NRC 39184.

This report also appears in *Theoretical Computer Science B: Logic, semantics and theory of programming*, vol. 160, June 1996.

Abstract

An operational model of nondeterministic processes coupled with a novel theory of divergence is presented. The operational model represents internal nondeterminism without using explicit internal transitions. Here the notion of internal state effectively replaces the familiar notion of internal transition, giving rise to an alternative operational view of processes: the weak process. Roughly, a weak process is a collection of stable internal states together with a set of transitions each of which is defined from an internal state to another weak process. Internal nondeterminism arises from such refinement of processes into multiple internal states.

A simple extension to the basic weak process model gives rise to an elaborate operational theory of divergence. According to this theory, the ability of a process to undertake an infinite internal computation which is pathological, or persistent, is distinguished from its ability to undertake an infinite internal computation which is not. Although applicable to process algebraic languages with an internal action construct, the resulting model is most suitable for supplying operational semantics to process algebras which express internal nondeterminism by an internal choice construct. The distinction between the two forms of divergence is in particular taken into account when the hiding construct of such a process algebra is assigned a weak process semantics.

1 Introduction

This paper presents *Abstract Transition Systems* (ATSs), an operational model of nondeterministic processes. A refinement of the basic *Labeled Transition System* (LTS) model [17], ATSs represents internal nondeterminism without using explicit internal transitions. In the ATS model, the familiar notion of an internal transition is replaced by that of an internal state, leading to an alternative view of processes: the *weak process*.

Although the concept of weak process is not as powerful as the concept of a process provided by LTSs extended with internal transitions, the ATS model can handle nondeterminism quite effectively. This lead us to suspect that the discriminating power achieved by incorporating explicit internal transitions to the basic LTS model might be unnecessary, even undesirable. A similar belief appears to dominate several existing semantic theories which support internal transitions as a modeling mechanism on the language level, but attempt to compensate for their presence on the semantic level.

The ATS model, in its basic form, falls short of a satisfactory treatment of divergence. This problem is remedied by a simple extension: *divergence sets*. The extension gives rise to a view of divergence which is more elaborate than the ones found in the literature. The application of the extended ATS model in assigning operational semantics to the likes of LOTOS's problematic hiding construct [19] is particularly interesting.

1.1 Background and Motivation

Several process algebraic languages such as CCS [20], ACP [3], and LOTOS [27], provide a special prefix construct to express the notion of internal computation, or transition. It is assumed that the environment can neither control the undertaking of such computations nor observe them. In CCS and ACP, this construct is the infamous τ. In LOTOS, it is the construct "$;\mathbf{i}$".

Although internal transitions themselves are invisible to the external environment, their ultimate effect on the external behavior of a system in which they occur may be detectable. Another class of process algebras capture directly this effect: *internal nondeterminism.* Examples are TCSP [5], the τ-less version of CCS proposed in [9], and the process algebra described in [14]. All of these languages — in place of a τ-like construct — supply an *internal choice* construct. This in turn gives rise to a more abstract and cleaner treatment of nondeterminism on the language level.

Let us consider CCS. In CCS, the constructs $+$ (the general purpose nondeterministic choice) and τ play a fundamental role. However, they have been found unsatisfactory by De Nicola and Hennessy, who suggested in [9] that they be replaced by two new constructs: $[]$ which models external nondeterminism and $\oplus$ which models internal nondeterminism. We quote from [9] adapting to the notation used in this paper:

> The semantic equivalence used in [20] and [8] are not preserved by $+$; they are not congruences. Also, $+$ exhibits a rather complicated mixture of intuitively different forms of nondeterministic behavior, often referred to as *internal* and *external* nondeterminism, see [23]. In the process $a{\cdot}P + b{\cdot}Q$, there is external nondeterminism: if the user requests an a synchronization, the process will oblige and subsequently act like P, whereas if b is requested, it will also be performed and the process will continue as Q. Internal nondeterminism is exhibited in $a{\cdot}P + a{\cdot}Q$ and $a{\cdot}P + \tau{\cdot}Q$. In the first cases the process will oblige when asked to perform an a synchronization but the user will have no control over which of P or Q the process will evolve to. The behavior of the process such as $a{\cdot}P + \tau{\cdot}Q$ is difficult to describe and the special symbol τ to represent internal actions is counterintuitive; if the actions are internal and invisible, there should be no need to refer to them in the language or calculus. The laws governing the manipulation of τ in the calculus [15] are rather mysterious and to date nobody has been successful in providing an intuitive and acceptable model which explains the nature of τ.

De Nicola and Hennessy show that a better treatment of nondeterminism is possible without introducing internal transitions on the language level. Ironically, the operational semantics that they provide for this τ-less version of CCS is in reality not free of the concept of internal transition, although technically the "unlabeled" transition relation they employ to model "invisible moves" is different from $-\tau\rightarrow$ used in the original CCS. The behavioral equivalence that they define later abstracts from these so-called "invisible moves". This paradox is common in traditional process algebra: the behavioral equivalence, not the operational semantics of the relevant constructs, is mainly responsible for the abstraction of internal behavior. We raise the following question:

> *If it is desirable to omit internal transitions on both the language and the semantic level, then can an explicit notion of internal transition be avoided altogether?*

We show in this paper that the answer to the above question is positive. If a semantic theory should ultimately abstract from internal transitions, then the discriminating power gained by incorporating an explicit notion of internal transition to the underlying model seems useful only as a notational convenience. In fact, several existing denotational theories successfully avoid referring to internal transitions while providing a satisfactory treatment of nondeterminism; to name a few, we can cite acceptance trees [14], the Improved Failures Model [6], rooted failure trees [19], and the Readiness

Model [23].[1] Implicit to all of these extended trace models is a notion less powerful than that of an internal transition: the *internal state*. In his account of common semantic theories [28], Glabbeek introduces several types of abstract machines on which processes under observation are assumed to run. Each type of machine justifies a particular type of semantics from a testing perspective. The notion of internal state also seems to be the underlying common concept in these machines. The main purpose of this paper is to investigate the nature of this implicit notion and give it an operational identity.

Introduced by Keller in his 1976 article [17], Labeled Transition Systems provide a convenient method for describing the step-by-step behavior of a reactive system. As such, the LTS model supplies a common basis for studying the interrelations between several existing operational theories of concurrency and nondeterminism (for examples, see [7], [12], and [19]). In particular, it has been adopted as the underlying operational model for CCS and LOTOS. In the heart of a LTS is a transition relation defined on a set of processes. The transition relation describes how these processes evolve through performing elementary actions drawn from a fixed set. To model nondeterminism more effectively, the basic LTS model is often endowed with a special action — the operational analogue of the CCS internal action construct τ — which represents a one-step internal computation.

The operational semantics of a process algebraic language is often specified by means of a set of inference rules which permit one to build a corresponding LTS, or another form of abstract machine, from a given expression of the language. Thus the inference rules collectively can be thought of as defining a mapping from the set of all legal expressions to the processes of a very large, or universal, LTS. This approach, called structured operational semantics, has been advocated as a general method for assigning formal semantics to programming languages [25, 22]. Since CCS [20], it has been used widely in the process algebraic framework. For process algebras, the resulting semantics is usually equipped with a behavioral equivalence relation between processes, providing an abstract notion of external behavior. This equivalence, among other things, usually takes into account the presence of internal transitions — i.e., try to capture their ultimate effect on the external behavior of the processes in which they appear. On the one hand, the behavioral equivalence may be characterized in three different ways: (1) *structurally*, i.e., relying entirely on the operational structure of processes, (2) *logically*, i.e., in terms of a modal logic, or (3) with respect to a particular perspective of external *testing*. On the other hand, it may be *direct* or induced *indirectly* by a *preorder* (transitive and reflexive relation).

There are numerous behavioral equivalences defined on the structure of an LTS. Examples can be found in [8], [7], [19], and [18]. Perhaps the most well-known of

[1]These models are often referred to as extended trace theories.

all is Milner's observation equivalence [21]. The original definition of observation equivalence was inductive. Later, it was given a more workable formulation based on Park's notion of bisimulation [24]. Park's formulation had an impact on the way behavioral relations were defined: several other researchers that followed used similar formulations in a variety of frameworks [18, 29, 10].

Although most existing denotational theories of concurrency do not require an explicit notion of internal action, or transition, for a satisfactory treatment of nondeterminism, the literature lacks an effective operational model which is free of explicit internal transitions. An internal choice construct does not have a natural translation in the LTS model: its operational semantics is often defined in terms of multiple internal transitions originating from a common process. What is therefore needed is a more abstract treatment of internal nondeterminism — an operational model in which the likes of TCSP's internal choice construct has a more natural translation. The ATS model [10] fills in this gap.

The basic ATS model does not provide a satisfactory treatment for *divergence* — the ability of a system to undertake an infinite internal computation. Divergence is often modeled operationally as the possible execution of an infinite sequence of internal transitions. It usually results from the sometimes pathological combination of internal nondeterminism, inter-process communication, and abstraction (hiding) of internal behavior. Different operational theories treat divergence differently, the adequacy of a particular treatment depending on the application. For example, for CCS, a partially satisfactory operational treatment is provided through observation and weak bisimulation equivalences [21], and more complete treatments have been proposed by Walker in terms of preorder-based refinements of the weak bisimulation relation [29]. In some other frameworks, divergence is completely disregarded or sometimes equated with deadlock or termination. For example, the behavioral equivalence induced by the *may*-testing preorder [14] (also called strings or trace equivalence [7]) adopts this view. By contrast, the equivalence induced by the *must*-testing preorder [14, 8] and failures equivalence [6] always consider divergence as catastrophic. According to the catastrophic view, no distinction is made among divergent processes independent of their potential to also exhibit an external behavior during divergence. A compromise between these two extreme views has been achieved by a variation of the testing equivalence whose original version was proposed by Brinksma [4]. This latter equivalence — due to Leduc [19] — while being able to detect divergence of any form, rejects the catastrophic interpretation. For an early comparison of several different treatments of divergence with respect to the underlying behavioral equivalences, see [7].

An even more refined view would acknowledge that divergence occurs in different forms having different implications: some are regarded as pathological, whereas others are more acceptable. We assume that being able to distinguish between pathological and non-pathological forms of divergence is important. This is a perspective which other known operational theories fail to support. After the incorporation of divergence property sets, the ATS model also fills in this gap.

1.2 Outline

In Section 2, we introduce the relevant elementary concepts. First we give an informal definition of a process, followed by a discussion of the fundamental notions of communication, concurrency, and nondeterminism. The relationships among these notions are exploited, all from an intuitive perspective. Then we explain, in the context of an example, the two distinct forms of nondeterminism that concurrent systems may generally exhibit. By the end of Section 2, we will have introduced a simple syntax for describing concurrent nondeterministic processes. This syntax will be formalized and given a concrete operational semantics in Section 8.

In Section 3, Labeled Transition Systems are defined. This is followed in Section 4 by a discussion of internal transitions vs. internal states. Simultaneously, the notion of weak process is developed. Roughly, a weak process models a system whose external behavior is abstracted such that nondeterminism resulting from the internal computations of the system is encoded in terms of explicit internal states. An internal state may be thought of as capturing the ultimate effect, on the external behavior of a nondeterministic system, of a sequence of internal computations. Section 5 introduces Abstract Transition Systems as an operational model of weak processes, and Section 6 discusses their expressive power *vis-à-vis* the LTS model. We prove the basic ATS model to be less powerful than the LTS model with explicit internal transitions.

In Section 7, the basic ATS model presented in Section 5 is extended with divergence sets, resulting in a potentially more powerful framework which we call *Extended Abstract Transition Systems* (EATSs). This extension leads to an original, elaborate, and explicit treatment of divergence. According to this treatment, the ability of a process to undertake an internal computation during which evolution to an externally controllable behavior would not be possible past a certain point (called pathological or *strong* divergence) is distinguished from its ability to undertake an infinite internal computation during which evolution to an externally controllable behavior remains always possible (called non-pathological or *weak* divergence). At the end of the section, we state that even after the incorporation of divergence property sets, the ATS model has less discriminating power than the LTS model (with internal transitions), up to the strongest divergence discriminating behavioral equivalences imaginable for both models.

In Section 8, we turn the syntax introduced in Section 2 into a concrete process algebra, which we call MPA — a *Minimal Process Algebra.* To demonstrate the applicability of the EATS model, we provide a weak process semantics for MPA in terms of EATSs.

In Section 9, the difficulty with the weak process semantics of process algebras having an internal action construct is touched upon briefly in the context of LOTOS.

2 Basic Concepts

2.1 The Notion of a Process

Processes model dynamic systems. However, here our view of dynamic systems will be rather restricted — in that we assume such things as the priorities, durations, starting times, and ending times of events, and the actual structure of the information exchanged between functional components are not crucial for the understanding of the systems under investigation.

In this view, one may think of a process as an abstract machine which performs *actions* in some prescribed manner. The most important assumption made here is that these actions are indivisible, or *atomic*. In general, the actions are uninterpreted, although to enhance understanding, they may be interpreted as commands, instructions, events, signals, messages, or communication primitives, depending on the context.

A process is executed as follows: Choose an initial action; perform it and see which actions are offered next. Then pick a second action; perform it and see which actions are offered next, and so on. *When it is possible for a process to perform a particular action, we say that the process* offers *that action.*

This summarizes our perspective of a process in its most general form.

2.2 Inter-Process Communication

Processes are not very useful without the ability to interact, or communicate, with each other. By *communication*, we refer to the ability of a process to constrain or influence the behavior of another process. Here we assume that this ability is exercised when two or more processes perform a reserved action simultaneously. Such an action may only be performed when all the participants offer that action at the same instant. This regime of process interaction is often referred to as *synchronous communication* (also known as *rendez-vous*). As do many other models and languages [16, 20, 5, 3, 11, 14, 27], we combine synchronous communication with *asynchronous evolution*, i.e, assume that communicating processes evolve at independent speeds.

2.3 Concurrency and Nondeterminism

Let P be a process which offers the action a, and then terminates. Similarly, let Q be a process which offers the action b, and then terminates. We express P and Q as

$$\begin{array}{lll} P & \textbf{def} & a \cdot NIL, \\ Q & \textbf{def} & b \cdot NIL, \end{array}$$

where NIL indicates termination. Suppose these two processes are completely independent; i.e., there is no communication between them. If desired, P and Q may be interpreted as two independent sequential programs, with a and b representing input statements. In addition, we assume that at any given instant, only one action may be performed; for example, as would be the case if P and Q time-shared a central processing unit.

Now suppose we execute P and Q concurrently and observe the resulting behavior, which we denote by $P\langle\rangle Q$. The execution of $P\langle\rangle Q$ may be initiated by picking an action from P or an action from Q. Let us start with P: so we let P perform a, and then Q perform b. This particular scenario is represented by the *execution trace* $a \cdot b$. Yet, this is only one of the two possible scenarios; the other is represented by the trace $b \cdot a$. As such, the behavior of $P\langle\rangle Q$ can be described explicitly by gathering all possible interleavings of the individual execution traces of P and Q. We express this by writing:

$$a \cdot b \cdot NIL + b \cdot a \cdot NIL.$$

In effect, $P\langle\rangle Q$ can be thought of as a *nondeterministic* process, where the order in which the actions a and b are executed — although externally controllable may it be — is unknown *a priori*. Therefore, concurrency is reduced to nondeterministic interleaving of atomic actions.

2.4 Internal and External Nondeterminism

We distinguish between two fundamental forms of nondeterminism. The first form describes a behavior which may be influenced externally. Here the environment — rather than the system itself — decides which action to perform next, although it is the system which presents the alternatives to the environment. This form of nondeterminism is referred to as *external nondeterminism*. The other form is *internal nondeterminism*, where the system makes an internal decision about its subsequent behavior and the environment cannot interfere with this decision.

As an example, consider a system which repeatedly performs the action a until it receives a time-out signal. The time-out signal is modeled by the action t. When the system receives the time-out, it offers the action b instead of the action a, and then recycles. We may describe this system by a recursive process:

$$P \textbf{ def } a{\cdot}P + t{\cdot}b{\cdot}P.$$

In the above expression, + indicates an external nondeterministic choice between the sub-behaviors $a{\cdot}P$ (perform a and recycle) and $t{\cdot}b{\cdot}P$ (perform t followed by b, and then recycle.) Therefore, initially it is possible for P to perform both the action a and the action t, the choice belonging to the environment. The following is a typical execution trace of P:

$$a{\cdot}a{\cdot}t{\cdot}b{\cdot}a{\cdot}a{\cdot}a{\cdot}a{\cdot}t{\cdot}b{\cdot}a{\cdot}a{\cdot}\cdots$$

Now introduce a second process, Q, which models a timer, the source of the time-out signals. The process Q repeatedly performs the action t:

$$Q \textbf{ def } t{\cdot}Q.$$

Let us assume that the processes P and Q are executed concurrently — but unlike in the previous example, this time they communicate via the synchronization action t which, upon communication, becomes an internal event. We express the resulting behavior by

$$(P\langle t\rangle Q)\,\textbf{hide}\,\{t\}.$$

Here $P\langle t\rangle Q$ expresses the concurrent behavior of P and Q when the two processes communicate upon synchronizations on the action t. The abstraction of the action t in $P\langle t\rangle Q$ is expressed by the construct $\textbf{hide}\{t\}$. This may be better explained by a physical analogy:

Suppose a *gate* is associated with each possible action of a process and that all actions are performed at these gates. Looking from outside inwards, one can only observe actions performed at these gates. Returning to our time-out example, the external observer would then perceive the processes P and Q as two *black boxes*, P with three gates labeled a, b, and t, and Q with a single gate labeled t. This is depicted in Figure 1(a). Let us describe how to build the black box representing the process $(P\langle t\rangle Q)\,\textbf{hide}\,\{t\}$ from the black boxes of P and Q: First connect the t gate of P to the t gate of Q. Then place both black boxes in a new black box named $P\langle t\rangle Q$ with the gates a, b, and t. The gates of the bigger black box are connected to the matching gates of P and Q. Note that the t gate of the new black box is connected to both P and Q. We obtain the box depicted in Figure 1(b). If, subsequently, the t gate of the enclosing box $P\langle t\rangle Q$ is eliminated, we obtain the black box illustrated in Figure 1(c). This latter black box represents the process $(P\langle t\rangle Q)\,\textbf{hide}\,\{t\}$. Here the interactions at the internal gate t are not visible from the point of view of an external observer because we removed its external interface.

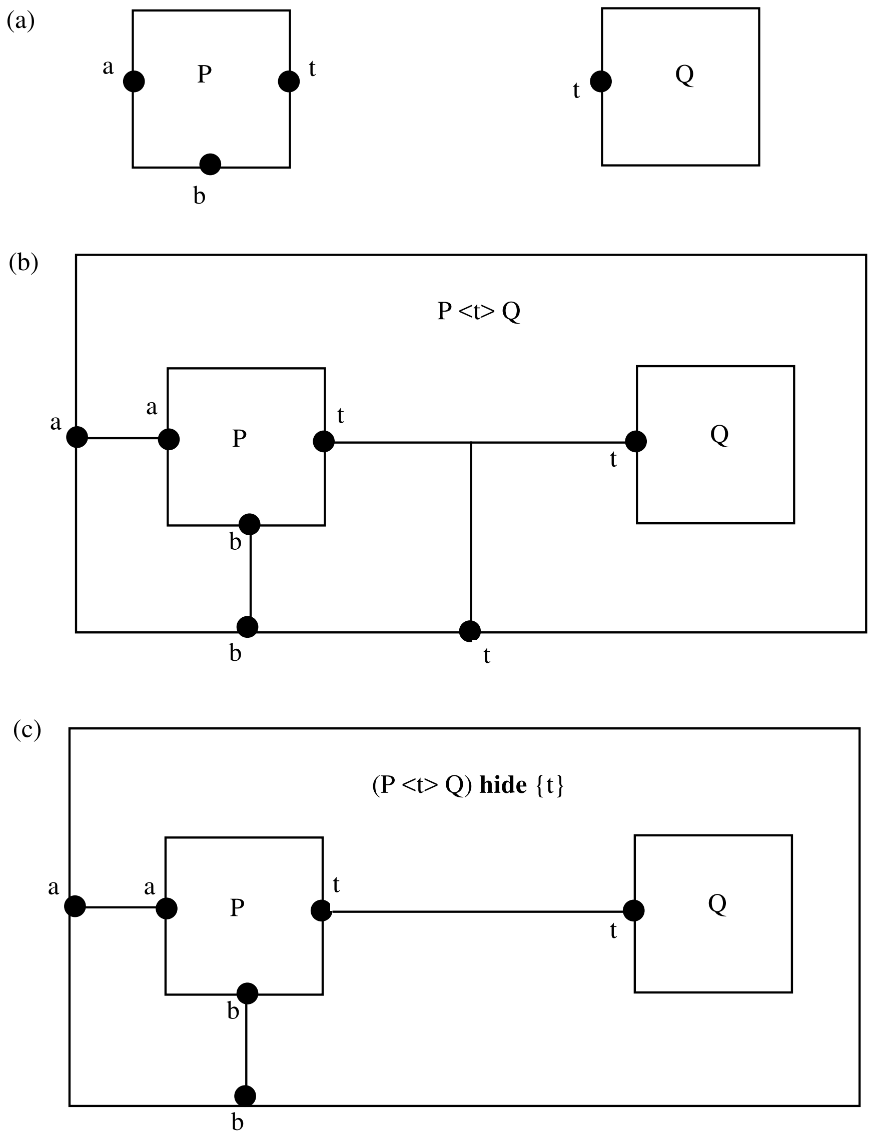


Figure 1: Black box view of parallel composition. (a) Black boxes representing the processes P and Q. (b) The black box representing the process $P\langle t\rangle Q$. (c) The black box representing the process $(P\langle t\rangle Q)$ **hide** $\{t\}$.

With the above interpretation in mind, let

$$R = (P\langle t\rangle Q)\ \mathbf{hide}\ \{t\}.$$

How can one describe R explicitly by reducing it to nondeterministic interleaving of its externally visible actions? As a first attempt, we may write R as

$$R\ \mathbf{def}\ a{\cdot}R + b{\cdot}R \tag{1}$$

Yet, this is not quite correct. Although it is possible for R to perform the action a, this is not always guaranteed. For instance, R may have just performed a t internally and be ready to perform only a b action. To be able to take account of this distinction, we write R as

$$R\ \mathbf{def}\ a{\cdot}R \oplus b{\cdot}R \tag{2}$$

where $\oplus$ expresses an *internal choice*. The correct interpretation is that R initially makes an internal decision about whether to behave as $a{\cdot}R$ or $b{\cdot}R$, and once this choice has been made, either only an a or only a b is offered to the environment. Having performed one of these actions, R recycles just as P and Q do. In (1) above, the environment is guaranteed success each time it chooses one of a or b to be performed, whereas in (2), it is not. The construct $+$ models external nondeterminism — or the form of nondeterminism which is externally controllable. By contrast, the construct $\oplus$ models internal nondeterminism — or the form of nondeterminism which is *not* externally controllable.

3 Labeled Transition Systems

Operationally, the ability of a process to evolve through performing a particular action a can be represented by a transition relation, denoted by $-a\rightarrow$. If the process P evolves to the process Q upon performing a, we write: $P{-a\rightarrow}Q$. Labeled Transition Systems capture precisely this intuition.

Definition 3.1 A *Labeled Transition System* (LTS) is a triple

$$\langle \mathcal{S}, Act, -\cdot\rightarrow \rangle,$$

where

i. $\mathcal{S}$ is a set of *strong processes*,

ii. Act is a (nonempty) set of atomic, external *actions*,

iii. $-\cdot\rightarrow\ \subseteq \mathcal{S} \times (Act \cup \{\tau\}) \times \mathcal{S}$ is a *transition relation*, and

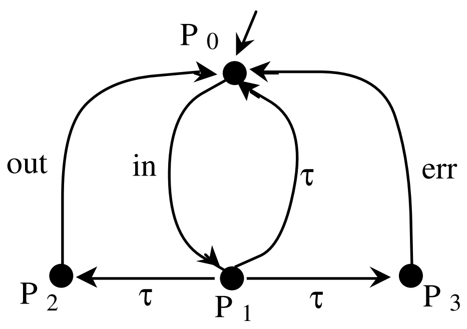


Figure 2: Transition graph of unreliable communication medium. The little arrow marks the initial behavior, or state, of the medium.

iv. τ, where $\tau \notin Act$, is a special action, called the *internal action.*

Here the term *strong* is used to imply a process specified by a LTS. In the upcoming sections, Abstract Transition Systems will be introduced. This model will be proved to be less powerful than LTSs. To make the distinction between the two models clear, we will call a process *weak* if it is specified in terms of the latter model.

Unlike an action in Act, the special action τ represents an internal computation which is neither directly observable nor externally controllable. This gives LTSs extra power for modeling various forms of nondeterminism, and also the phenomenon of divergence. Let us illustrate this point by a simple example.

Suppose we wish to model an unreliable communication medium which may (nondeterministically) deliver, corrupt, or lose messages. We assume that the medium has a maximum capacity of one message. The relevant external actions are: *in*, represents reception of a new message; *out*, represents delivery of an uncorrupted message; and *err*, represents delivery of an erroneous or corrupted message. The behavior of the medium can be described by a LTS whose *transition graph* is given in Figure 2. In the figure, the node (or process) P_0 models the (initial) behavior of the medium. The corresponding LTS is $\langle\{P_0, P_1, P_2, P_3\}, \{in, out, err\}, -\cdot\rightarrow\rangle$, where the transition relation $-\cdot\rightarrow$ is determined by the arcs (transitions) of the graph. The informal interpretation of this LTS is as follows:

- P_0 initially receives a message from the environment through performing the action *in* and then evolves to the process P_1.
- P_1 is inherently nondeterministic. Here there are three possibilities, none of which can be influenced externally. These are:
  1. The medium loses the current message and becomes ready to receive a new message. This is represented by the *internal transition* $P_1 - \tau \rightarrow P_0$.

Therefore, the loss of a message is modeled by the ability of the medium to accept a new message without delivering the previously received one.

2. The medium delivers the current message correctly and becomes ready to receive a new message. This is represented by the sequence of transitions $P_1 - \tau \rightarrow P_2 - out \rightarrow P_0$.
3. The medium manages to deliver the current message but the message is corrupted during transmission. Subsequently, it becomes ready to accept a new message. This is represented by the sequence of transitions $P_1 - \tau \rightarrow P_3 - err \rightarrow P_0$.

Note that the process P_1 — because of the presence of internal transitions — acts autonomously in making the decision whether to deliver, corrupt, or lose the current message.

The following general conventions will be used throughout the text:

Notation 3.2 Let A be an arbitrary set.

1. *Nat* denotes the set of natural numbers.
2. $\wp(A)$ denotes the powerset of A.
3. A^* denotes the set of all finite sequences over A.
4. $\varepsilon \in A^*$ denotes the empty sequence.
5. $\alpha^0 = \varepsilon$ for every $\alpha \in A$.
6. $\alpha^k \in A^*$, where $\alpha \in A$ and $k \in Nat$, denotes the finite sequence $\underbrace{\alpha\alpha\cdots\alpha}_{k \text{ times}}$.

We will take advantage of indexing functions to represent infinite sequences over a given set:

Definition 3.3 Let A be an arbitrary set. A function ρ from *Nat* to A is called an *indexing function* over A. For $i \in Nat$, we abbreviate $\rho(i)$ by ρ_i.

The following notation is adopted for LTSs:

Notation 3.4 Let $P, Q \in \mathcal{S}$; $a \in Act$; $\alpha \in Act \cup \{\tau\}$; and $\bar{\alpha} \in (Act \cup \{\tau\})^*$ be a nonempty finite sequence of internal or external actions such that $\bar{\alpha} = \alpha_1\alpha_2\cdots\alpha_n$ for $n > 0$. The α_i are not necessarily distinct. Let ρ be an indexing function over $Act \cup \{\tau\}$.

1. We write $P{-}\alpha{\rightarrow}Q$ whenever $\langle P, \alpha, Q\rangle \in {-}{\cdot}{\rightarrow}$.

2. $P{-}\varepsilon{\rightarrow}P$ is always true.

3. $P{-}\alpha{\rightarrow}$ means there exists $P' \in \mathcal{S}$ such that $P{-}\alpha{\rightarrow}P'$.

4. $P{\not-}\alpha{\rightarrow}$ means *not* $P{-}\alpha{\rightarrow}$.

5. $P{-}\bar{\alpha}{\rightarrow}Q$ means there exist $P_1, P_2, \ldots, P_{n-1} \in \mathcal{S}$ such that $P{-}\alpha_1{\rightarrow}P_1{-}\alpha_2{\rightarrow}\cdots{-}\alpha_{n-1}{\rightarrow}P_{n-1}{-}\alpha_n{\rightarrow}Q$.

6. $P{\not-}\bar{\alpha}{\rightarrow}$ means *not* $P{-}\bar{\alpha}{\rightarrow}$.

7. $P{-}\rho{\rightarrow}$ means there exists an indexing function Θ over $\mathcal{S}$ such that $\Theta_0 = P$ and for every $i \in \mathit{Nat}$, we have $\Theta_i{-}\rho_i{\rightarrow}\Theta_{i+1}$.

8. ${\not-}\rho{\rightarrow}$ means *not* ${-}\rho{\rightarrow}$.

We conclude this section by three relevant definitions:

Definition 3.5 Let $P \in \mathcal{S}$ be a strong process in a LTS.

1. P is called *stable* if $P{\not-}\tau{\rightarrow}$. Otherwise, it is called *unstable*.

2. P is called *terminal* if for every $\alpha \in \mathit{Act} \cup \{\tau\}$, we have $P{\not-}\alpha{\rightarrow}$.

3. $\mathit{ID}(P)$, called the *internal derivatives* of P, is the set of processes which are reachable from P via a finite uninterrupted sequence of internal transitions and which are either terminal or offer at least one external action:

$$\begin{aligned}\mathit{ID}(P) \overset{\text{def}}{=} \{\ Q \in \mathcal{S} : P{-}\tau^n{\rightarrow}Q \text{ for some } n \in \mathit{Nat} \text{ such that}\\ \text{either } Q{-}a{\rightarrow} \text{ for some } a \in \mathit{Act} \text{ or } Q \text{ is terminal }\}.\end{aligned}$$

4 The Concept of Weak Process

The presence of τ gives considerable discriminating power to the LTS model in terms of representing nondeterminism and divergence. When τ is allowed, it is fairly easy to construct a process whose behavior is rather difficult to understand. An example is the process whose transition graph is depicted in Figure 3.

Is it then possible to avoid internal transitions altogether and still be able to handle nondeterminism effectively? This question was put forward by Hennessy and De Nicola in [9] where the authors proposed to replace the cumbersome τ construct

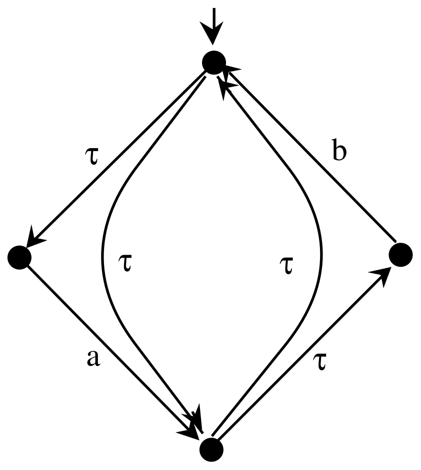


Figure 3: A strong process with internal transitions.

of CCS by the more convenient internal choice construct $\oplus$. We now investigate how this can be accomplished from a purely operational viewpoint.

From a black box perspective, it is possible to relate internal nondeterminism to the presence of more than one internal state. Consider a system which is known to be in a particular *unstable* state from which it may internally evolve to another unstable state, and subsequently to another one, and so on — without the environment being able to observe or control these spontaneous changes in behavior. If one abstracts time, it is possible to think of the undertaking of a finite sequence of such internal transitions in terms of a discrete internal decision that the system makes in an unstable state. Gathering all such possible decisions and disregarding the branching structure of the internal transitions, we may view the system in question as one which — having reached a particular *macro* state — chooses one of the several *internal states* associated with that macro state. Once an internal state has been chosen, the internal nondeterminism associated with the unstable macro state is assumed to be resolved. Therefore, macro states may be unstable when they encapsulate more than one internal state, whereas internal states are stable. Having chosen an internal state, the system may perform an external action and evolve to a new macro state where it may once again be faced with choosing among a new set of internal states.

In this light, it is conceivable that the notion of an internal transition can be replaced by that of an internal state, giving rise to a more abstract black box view of systems. This view is the foundation of the so-called *weak processes*. As the name suggests, the concept of a weak process is less powerful than the concept of a process provided by LTSs. Nevertheless, as we will demonstrate, it is effective enough for treating nondeterminism. Let us illustrate this idea in the context of a few examples.

Consider the transition graph of a strong process shown in Figure 4(a). We assume

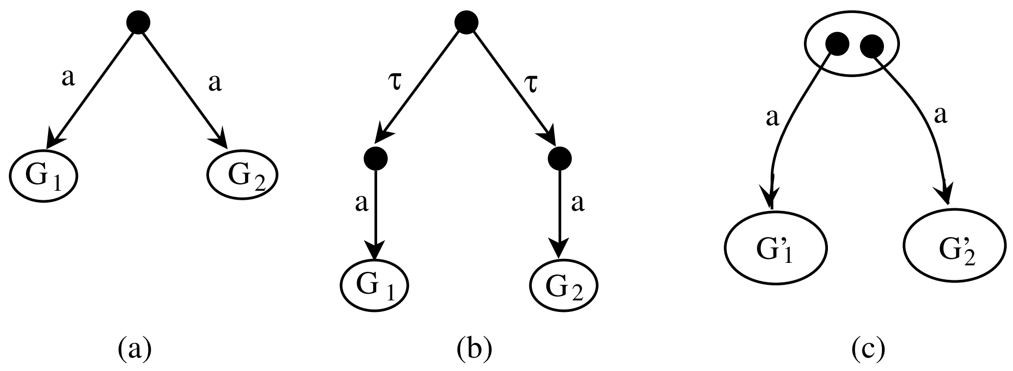


Figure 4: Transition graphs of an internally nondeterministic process.

that the subgraphs G_1 and G_2 are different. The usual interpretation of this transition graph is as follows: The root node represents a process which always offers the action a; however, prior to performing a, it makes an internal decision whether to evolve to G_1 or to G_2. From an external point of view, we may think of this process as having two internal states, each of which offers the action a, but the two evolve to different processes. We would not alter the intended external behavior if we replaced the transition graph of Figure 4(a) with that of Figure 4(b). Here the internal transitions model explicitly the internal choice of the process. We can in turn view the root of this latter graph as a kind of a *macro state*. Then each τ-reachable node of the root would correspond to a different internal state. This view leads to the weak process representation shown in Figure 4(c), where G'_1 and G'_2 are subgraphs corresponding to G_1 and G_2, respectively. In this new kind of transition graph,

i. big circles represent macro states, or *weak processes*,

ii. dark circles encapsulated in weak processes represent *internal states*,

iii. transitions are from internal states to weak processes, and

iv. each internal states has a deterministic branching structure.

Point (iv) above means that we regard an internal state as the primitive mechanism for modeling internal nondeterminism; i.e., an internal state cannot be further refined.

Figure 5 illustrates the correspondence between the two kinds of transition graphs of an internally deterministic process. Note that the graph on the right has a single internal state, implying an absence of internal nondeterminism.

Let us now consider a less obvious case, shown in Figure 6, where nondeterminism is not symmetric. This mixing of internal and external nondeterminism introduces a third kind of nondeterministic behavior which will not be catered for. Instead, we

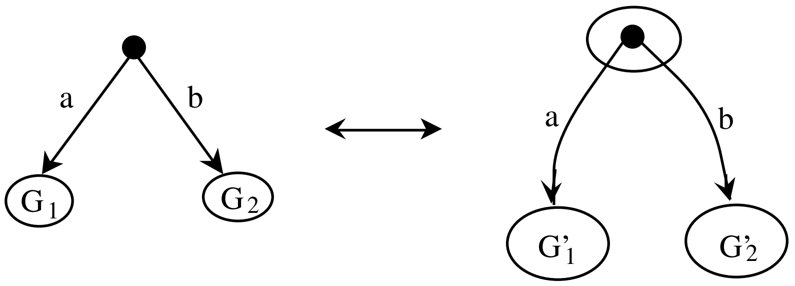


Figure 5: Transition graphs of an internally deterministic process.

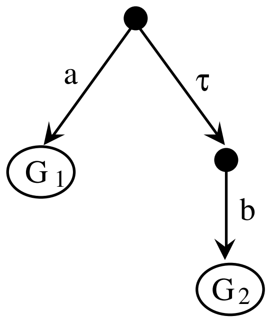


Figure 6: Transition graph of strong process with asymmetric nondeterminism.

will give two different interpretations of this transition graph in terms of its reduction to pure internal nondeterminism. To do this, we assume an environment consisting of an external observer who performs experiments on the process represented by the transition graph and registers the outcome. The experiments performed by the observer can be of type a or of type b. If the external observer performs an a experiment at a point when the action a is offered, we say that the action a is *accepted* and the process evolves to its subsequent state. If the process does not offer the action a at that point then we say that a is *refused*. Similarly for any other observable action.

The first interpretation reflects the traditional viewpoint, based on the "eventual possibility" of τ-reachable actions. According to this interpretation, the asymmetric transition graph represents a process which may initially accept or refuse the action a. If an a experiment is not performed then the process "eventually" offers the action b. Therefore, here we assume that if the observer waits long enough before a b experiment, the τ transition will sooner or later be undertaken. Consequently, the observer, who records both refusals and acceptances, would conclude after several experiments that

I. b is always (eventually) offered whereas a is not; i.e., sometimes both a and b are accepted and other times b is accepted and a is refused.

To make the above observation, whereas the observer must wait "as long as it takes" before each b experiment, he does not do so before each a experiment (if he waits long enough before each a experiment, the τ transition will be undertaken each time and he would not be able to register the possibility of the action a.) In other words, the actions a and b receive different treatments, and hence, the resulting nondeterminism is not symmetric. However, the external observer can be aware of this contextual difference between the two actions only if he has means to distinguish between stable and unstable states and conduct his experiments accordingly. That is, the fact that the actions a and b are offered in different contexts (i.e., a in an unstable state and b in a stable state) must be known externally.

Now let us consider a second interpretation which rejects the ability of the external observer to detect instabilities. Therefore, this time the observer may not tailor an experiment based on the detection of an unstable state. To justify this perspective, we assume that the process under observation runs on a Glabbeek-style machine which is a variant of the *readiness machine* defined in [28]. Let us call it an *acceptance machine*. The interface of an acceptance machine consists of:

1. a lamp called an *idle lamp* which is lit only when the process running on the machine idles (i.e., when no action is currently being carried out by the process under observation),

2. a button for each observable action, and

3. a lamp for each observable action.

The process can idle only in terminal states or in states which offer at least one observable action. Each time the process idles, the lamps of all actions the process offers in that state are lit. At any time the idle lamp is lit, the observer can perform a desired experiment by depressing the appropriate button (for an experiment of type a, the observer presses the button labeled a, and similarly for b). However, he can only perform those experiments corresponding to the actions with lit lamps. Each time the process idles, the observer can decide whether to abort the current run or continue, and if he aborts the current run, the remaining behavior of the process is recorded by means of the labels of the lit lamps. Thus each run results in an "observation" consisting of a sequence of actions and a *ready set* (an account of the set of actions whose lamps are lit at the end of that observation). The observer must abort an observation if the process idles and none of the lamps are lit, and he must abort it after a finite number of experiments. Once the idle lamp is lit, all internal activity of the process under observation is suspended; i.e., the process blocks and is not allowed to change its current state autonomously. The differences between the acceptance machine and the readiness machine are the following:

- In the readiness machine, the machine itself chooses a finite execution path that is consistent with the current state of the process under observation, not the observer. The machine can idle only at the end of an observation, and recovery from an idle period is not possible.

- Instead of an idle lamp, the interface of the readiness machine is equipped with a display that shows the action currently carried out by the process under observation. This display is the means by which the observer records the execution trace of a run.

- The semantics of hiding in [23] suggests that in the readiness machine of Glabbeek, the process under observation can idle only in a stable state. In the acceptance machine, idling in an unstable state which offers at least one observable action is permitted. Therefore, in the readiness machine, the external observer can identify stable and unstable internal states, whereas in the acceptance machine, this is not possible. As a result, unlike with the readiness machine, instability resulting from asymmetric nondeterminism is undetectable with the acceptance machine.

Therefore, for the process depicted by the asymmetric transition graph of Figure 6, acceptance machine type observations would lead to the following conclusion:

II. sometimes a is accepted and b is refused and other times b is accepted and a is refused.

This latter view is founded on the fact that in the example considered, the actions a and b are never offered at the same time.

Let us represent the two suggested ways of interpreting asymmetric nondeterminism in terms of weak processes:

According to interpretation (I) above, there are two internal states; in one of them both a and b are offered whereas in the other, only b is possible. This is depicted in Figure 7. In (II) above, the asymmetry present in (I) is broken. In one of the internal states, only a is offered, whereas in the other, only b is offered. This is illustrated in Figure 8.

In this paper, we adopt interpretation (II) for two reasons: First, it is less demanding in terms of the required capabilities of the external observer. Second, by suggesting a one-to-one correspondence between internal states of a weak process and the internal derivatives of a related strong process, interpretation (II) leads to substantial simplifications both in establishing the expressiveness result of Section 6 and in giving semantics to the hiding construct of the example process algebra. Nonetheless, it should be pointed out that the expressiveness result of Section 6 is independent of the particular interpretation adopted, and the suggested semantics

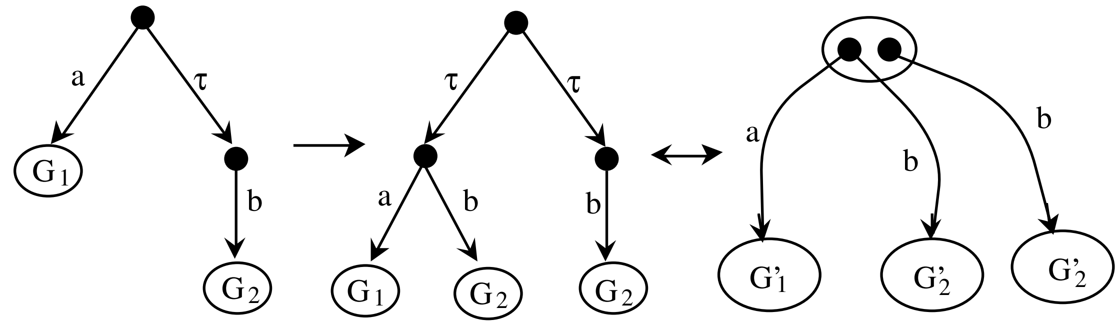


Figure 7: Asymmetric nondeterminism — Interpretation (I).

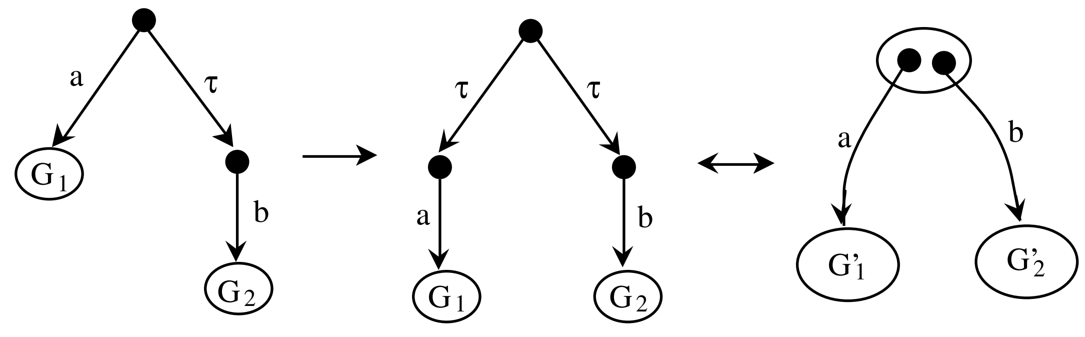


Figure 8: Asymmetric nondeterminism — Interpretation (II).

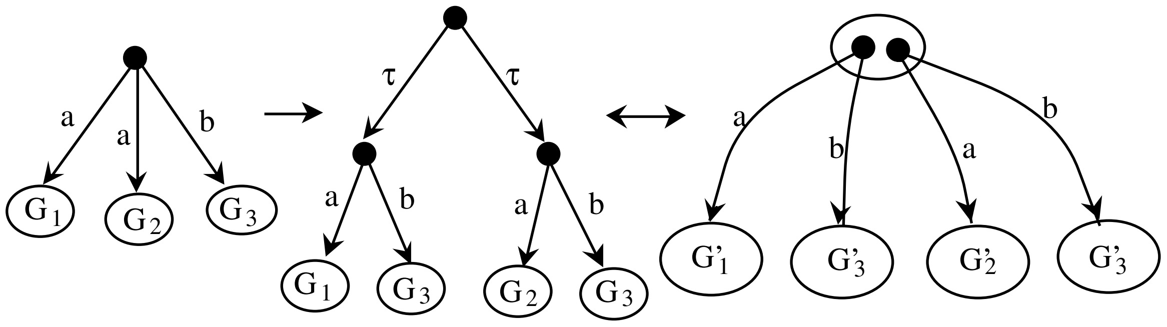


Figure 9: Transition graphs of a process which is nondeterministic in a and deterministic in b.

of the hiding construct can easily be modified to conform to interpretation (I).[2] It should be pointed out that observation equivalence [21, 1], testing equivalence [4, 19], failure equivalence [6] and *must*-testing equivalence [8, 14] are all based upon a form of external testing which is compatible with interpretation (I) rather than with interpretation (II); so the view adopted here unfortunately represents a deviation from the popular one.

Before concluding this subsection, let us consider a last example, shown in Figure 9, in which the system considered is deterministic with respect to the action b but nondeterministic with respect to the action a. It is important to note that b is never refused and therefore, it appears in both internal states. If instead of being directly accessible, b were accessible through an internal action, then we would have three internal states. This latter situation is depicted in Figure 10.

[2]As an objection to interpretation (II), it has been pointed out that one can construct contexts that distinguish the LOTOS processes

$$Q = a; \mathbf{stop}[]\mathbf{i}; b; \mathbf{stop} \quad \text{and} \quad R = \mathbf{i}; a; \mathbf{stop}[]\mathbf{i}; b; \mathbf{stop}$$

which one can express by the expressions $(a{\cdot}NIL{+}d{\cdot}b{\cdot}NIL)\mathbf{hide}\{d\}$ and $(d{\cdot}a{\cdot}NIL{+}d{\cdot}b{\cdot}NIL)\mathbf{hide}\{d\}$, respectively. One such context is

$$\lambda P.P|[a,b]|(b; c; \mathbf{stop})$$

or $\lambda P.P\langle\{a,b\}\rangle(b{\cdot}c{\cdot}NIL)$ in our notation. According to LOTOS semantics, the process Q never deadlocks in this context while the process R may. However, if one rejects the premise that instabilities are externally detectable, it would potentially be possible for an external observer to record an occasional deadlock in the composite process $Q|[a,b]|(b; c; \mathbf{stop})$ because of the experiments completed prior to the undertaking of the internal transition in Q. In other words, the process Q may deadlock in the given context with respect to acceptance machine observations.

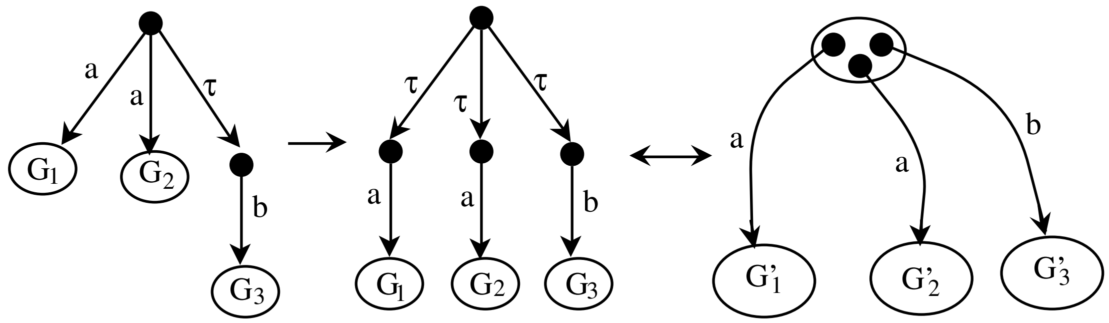


Figure 10: Transition graphs of a process which is nondeterministic both in a and in b.

5 Abstract Transition Systems

We now formally introduce an operational model in which internal nondeterminism is represented effectively without using explicit internal transitions. Thus we omit the τ action; instead, we allow processes to be refined into one or more internal states.

Definition 5.1 An *Abstract Transition System* (ATS) is a quintuple

$$\langle \mathcal{W}, \mathcal{I}, Act, Int, =\!\cdot\!\Rightarrow \rangle$$

where

i. $\mathcal{W}$ is a set of *weak processes*,

ii. $\mathcal{I}$ is a set of *internal states*,

iii. *Act* is a (nonempty) set of atomic, external *actions*,

iv. *Int* is a function from $\mathcal{W}$ to $\wp(\mathcal{I})$, and

v. $=\!\cdot\!\Rightarrow\ \subseteq\ \mathcal{I} \times Act \times \mathcal{W}$ is a ternary relation satisfying for all $s \in \mathcal{I}$, $a \in Act$, and $P, P' \in \mathcal{W}$: $\langle s, a, P\rangle \in\ =\!\cdot\!\Rightarrow$ and $\langle s, a, P'\rangle \in\ =\!\cdot\!\Rightarrow$ implies $P = P'$.

Note that by (v) above, internal states must be (internally) deterministic. The only way to represent internal nondeterminism is through associating multiple internal states with a weak process.

As an example, consider again the unreliable communication medium illustrated in Figure 2. This time we represent the medium by a *transition graph with internal states*, as shown in Figure 11. The graph corresponds to an ATS, and the (initial) behavior of the medium is modeled by the node (or weak process) Q_0.

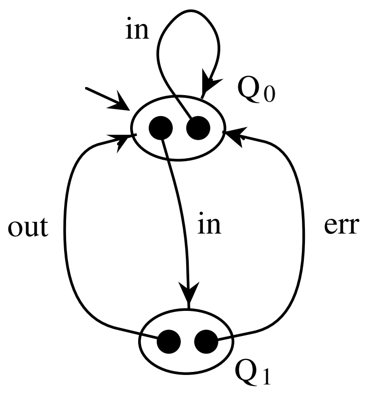


Figure 11: Transition graph of a weak process modeling the behavior of an unreliable communication medium.

Notation 5.2 Let $s \in \mathcal{I}$; $P, Q \in \mathcal{W}$; $a \in Act$; and $\bar{a} \in Act^* - \{\varepsilon\}$ be a nonempty finite sequence of actions such that $\bar{a} = a_1 a_2 \cdots a_n$ for $n > 0$. The a_i are not necessarily distinct. Let $\rho \colon Nat \longmapsto Act$ be an indexing function over Act.

1. We write $s{=}a{\Rightarrow}Q$ whenever $\langle s, a, Q\rangle \in {=}\cdot{\Rightarrow}$. This may be read as "the internal state s offers the action a; and upon performing this action, it evolves to the (weak) process Q."
2. We write $P[s]$ whenever $s \in Int(P)$.
3. $s{=}\bar{a}{\Rightarrow}Q$ means there exist $P_1, P_2, \ldots, P_{n-1} \in \mathcal{W}$ and $s_1, s_2, \ldots, s_{n-1} \in \mathcal{I}$ such that $s{=}a_1{\Rightarrow}P_1, P_1[s_1], s_1{=}a_2{\Rightarrow}P_2, P_2[s_2], \ldots, P_{n-1}[s_{n-1}], s_{n-1}{=}a_n{\Rightarrow}Q$.
4. $P[s]{=}\bar{a}{\Rightarrow}Q$ is an abbreviation for "$P[s]$ and $s{=}\bar{a}{\Rightarrow}Q$."
5. $P{=}\varepsilon{\Rightarrow}P$ is always true.
6. $s{=}\rho{\Rightarrow}$ means there exist indexing functions Φ over $\mathcal{W}$ and σ over $\mathcal{I}$ such that $\sigma_0 = s$ and for every $i \in Nat$, we have $\sigma_i{=}\rho_i{\Rightarrow}\Phi_i$ with $\Phi_i[\sigma_{i+1}]$.
7. $s{\not=}\rho{\Rightarrow}$ means *not* $s{=}\rho{\Rightarrow}$.
8. $P{=}\bar{a}{\Rightarrow}Q$ means there exists $s \in Int(P)$ such that $s{=}\bar{a}{\Rightarrow}Q$.
9. $s{=}\bar{a}{\Rightarrow}$ means there exists $P' \in \mathcal{W}$ such that $s{=}\bar{a}{\Rightarrow}P'$.
10. $s{\not=}\bar{a}{\Rightarrow}$ means *not* $s{=}\bar{a}{\Rightarrow}$.
11. $P[s]{\not=}\bar{a}{\Rightarrow}$ means $P[s]$ and $s{\not=}\bar{a}{\Rightarrow}$.

The concepts of stability and termination — which were originally defined for strong processes — have their weak process counterparts in the ATS model:

Definition 5.3 Let $P \in \mathcal{W}$ be a weak process in an ATS.

1. P is *stable* if P has exactly one internal state.
2. An internal state s is called *terminal* if for every $a \in Act$, $s \not\xRightarrow{a}$. The process P is terminal if $Int(P) \neq \emptyset$ and all of P's internal states are terminal.

A stable weak process is one which does not exhibit internal nondeterminism.

6 Expressive Power of Weak Processes

It is inevitable that some information will be lost in switching from strong to weak processes. We now substantiate this claim by finding an isomorphism between the weak processes in a given ATS and a specific subset of the strong processes in a related LTS. This subset determines where the basic ATS model stands with respect to the more conventional LTS model. The isomorphism is due to two interrelated behavioral equivalences, one defined on weak processes and the other on strong processes. Both of these equivalences are induced by bisimulations.

6.1 Equivalences on Weak and Strong Processes

The discussion of this section is centered on a new equivalence over *strong processes.* This equivalence — which is defined in view of Park's bisimulation [24] — is strictly weaker than the strong bisimulation equivalence discussed in [21].[3] The new equivalence will be referred to as *coarse equivalence. Coarse equivalence characterizes the set of weak processes as a subset of the set of strong processes.*

A second equivalence will be defined directly over weak processes. This latter equivalence will also be named *strong equivalence*, because it corresponds to the strongest of all relevant equivalences defined on the structure of an ATS. We are only interested in those equivalences on weak processes which are refined by strong equivalence.

We begin by the usual definition of strong (bisimulation) equivalence, i.e., the way it is defined for strong processes. This equivalence treats τ as any other action,

[3]It will be shown that this new equivalence is weaker than strong bisimulation equivalence. It also appears to be weaker than weak bisimulation equivalence [21] for strongly convergent processes. See [7] for the definition of strong convergence.

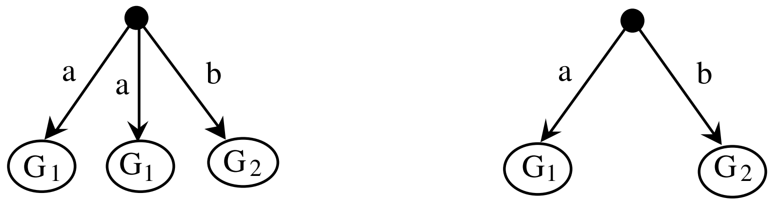


Figure 12: Two strong processes which are equated by strong bisimulation equivalence.

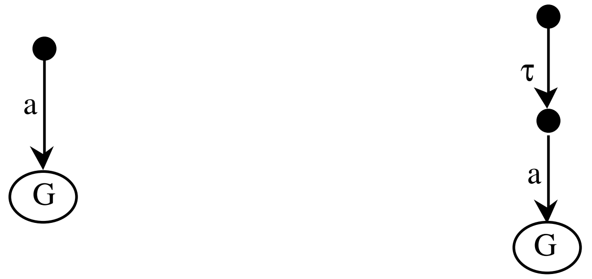


Figure 13: Two strong processes which are distinguished by strong bisimulation equivalence.

and consequently, it falls short of equating certain processes whose external behaviors would be indistinguishable for most practical purposes. For example, by strong equivalence, the processes depicted in Figure 12 are successfully equated; however, the processes depicted in Figure 13 are distinguished. In Figure 12, the process on the left hand side is only seemingly internally nondeterministic. In other words, the kind of internal nondeterminism exhibited by this process is not externally detectable. Therefore, any meaningful equivalence on strong processes should equate this process with the one on its right hand side.

Definition 6.1 (Milner [21]) A binary relation $\mathcal{R}$ on strong processes in a LTS is called a *strong bisimulation* if for all $P, Q \in \mathcal{S}$, $P \mathcal{R} Q$ implies for every $\alpha \in Act \cup \{\tau\}$ and for every $P', Q' \in \mathcal{S}$:

i. $P{-}\alpha{\rightarrow}P'$ implies for some $Q'' \in \mathcal{S}$, $Q{-}\alpha{\rightarrow}Q''$ and $P' \mathcal{R} Q''$.

ii. $Q{-}\alpha{\rightarrow}Q'$ implies for some $P'' \in \mathcal{S}$, $P{-}\alpha{\rightarrow}P''$ and $P'' \mathcal{R} Q'$.

Two strong processes P and Q are *strongly equivalent*, written $P \sim Q$, if there exists a strong bisimulation containing the pair $\langle P, Q \rangle$.

The proof of the following proposition can be found in [21]:

Proposition 6.2 $\sim$ *is an equivalence on the strong processes of a LTS.*

Now we define coarse bisimulation and the corresponding equivalence:

Definition 6.3 A binary relation $\mathcal{R}$ on strong processes in a LTS is called a *coarse bisimulation* if for all $P, Q \in \mathcal{S}$, $P \mathrel{\mathcal{R}} Q$ implies:

ii. For every $P' \in ID(P)$, there exists $Q' \in ID(Q)$ such that for all $a \in Act$ and for all $P'' \in \mathcal{S}$: $P' {-a\rightarrow} P''$ implies for some $Q'' \in \mathcal{S}$, $Q' {-a\rightarrow} Q''$ and $P'' \mathrel{\mathcal{R}} Q''$.

ii. For every $Q' \in ID(Q)$, there exists $P' \in ID(P)$ such that for all $a \in Act$ and for all $Q'' \in \mathcal{S}$: $Q' {-a\rightarrow} Q''$ implies for some $P'' \in \mathcal{S}$, $P' {-a\rightarrow} P''$ and $P'' \mathrel{\mathcal{R}} Q''$.

iii. P has a terminal internal derivative iff Q has a terminal internal derivative.

Two strong processes P and Q are *coarsely equivalent*, written $P \sim_c Q$, if there exists a coarse bisimulation containing the pair $\langle P, Q \rangle$.

Proposition 6.4 $\sim_c$ *is weaker than* $\sim$.

Proof Let $\mathcal{R}_s$ be a strong bisimulation and $P \mathrel{\mathcal{R}_s} Q$. $\mathcal{R}_s$ can easily be seen to satisfy clause (iii) of Definition 6.3. Now suppose $P' \in ID(P)$ and $P' {-a\rightarrow} P''$ for $a \in Act$. Then for some $n \in Nat$, we have

$$P {-\tau\rightarrow} P_1 {-\tau\rightarrow} P_2 {-\tau\rightarrow} \cdots {-\tau\rightarrow} P_{n-1} {-\tau\rightarrow} P'.$$

Therefore, we can find $Q_1, \ldots, Q_{n-1}, Q', Q'' \in \mathcal{S}$ satisfying $Q {-\tau\rightarrow} Q_1$ with $P_1 \mathrel{\mathcal{R}_s} Q_1$, $Q_1 {-\tau\rightarrow} Q_2$ with $P_2 \mathrel{\mathcal{R}_s} Q_2$, $\ldots$, $Q_{n-1} {-\tau\rightarrow} Q'$ with $P' \mathrel{\mathcal{R}_s} Q'$, and $Q' {-a\rightarrow} Q''$ with $P'' \mathrel{\mathcal{R}_s} Q''$. Since $Q' \in ID(Q)$, $\mathcal{R}_s$ satisfies clause (i) of Definition 6.3.

By similar reasoning, for every $Q' \in ID(Q)$ and for every $Q'' \in \mathcal{S}$ such that $Q' {-a\rightarrow} Q''$, where $a \in Act$, we can find $P' \in ID(P)$ and $P'' \in \mathcal{S}$ which satisfy $P' {-a\rightarrow} P''$ and $P'' \mathrel{\mathcal{R}_s} Q''$. Hence $\mathcal{R}_s$ satisfies clause (ii) of Definition 6.3. Consequently, $\mathcal{R}_s$ is a coarse bisimulation. $\square$

Proposition 6.5 $\sim_c$ *is an equivalence relation over strong processes.*

Proof We show that $\sim_c$ is (1) reflexive, (2) symmetric, and (3) transitive.

1. *Reflexivity*: Immediate from the definition of $\sim_c$.

2. *Symmetry*: Let $\mathcal{R}$ be a coarse bisimulation on $\mathcal{S}$. We consider $\mathcal{R}^{-1}$, the inverse of $\mathcal{R}$, which is defined as: $\mathcal{R}^{-1} \overset{\text{def}}{=} \{\langle x, y \rangle : y \mathrel{\mathcal{R}} x\}$. By the symmetry of Definition 6.3, $\mathcal{R}^{-1}$ is also a coarse bisimulation.

3. *Transitivity*: Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two coarse bisimulations on $\mathcal{S}$. We consider $\mathcal{R}_1\mathcal{R}_2$, the composition of $\mathcal{R}_1$ with $\mathcal{R}_2$, which is defined as:

$$\mathcal{R}_1\mathcal{R}_2 \overset{\text{def}}{=} \{\langle x, z\rangle : x\ \mathcal{R}_1\ y \text{ and } y\ \mathcal{R}_2\ z \text{ for some } y\}.$$

   Suppose $P_1\ \mathcal{R}_1\mathcal{R}_2\ P_2$. Then for some $P \in \mathcal{S}$, $P_1\ \mathcal{R}_1\ P$ and $P\ \mathcal{R}_2\ P_2$. Clearly, P_1 has a terminal internal derivative iff P_2 has a terminal internal derivative. Now let $P_1' \in ID(P_1)$ and $P_1'{-}a{\rightarrow}P_1''$. Then there exists $P' \in ID(P)$ such that for some P'', $P'{-}a{\rightarrow}P''$ with $P_1''\ \mathcal{R}_1\ P''$. Also, since $P\ \mathcal{R}_2\ P_2$, there exists $P_2' \in ID(P_2)$ such that for some $P_2'' \in \mathcal{S}$, $P_2'{-}a{\rightarrow}P_2''$ with $P''\ \mathcal{R}_2\ P_2''$. Therefore, $P_1''\ \mathcal{R}_1\mathcal{R}_2\ P_2''$. Similarly, we can also find for every $P_2' \in ID(P_2)$ satisfying $P_2'{-}a{\rightarrow}P_2''$, two processes $P_1' \in ID(P_1)$ and $P_1'' \in \mathcal{S}$ such that $P_1'{-}a{\rightarrow}P_1''$ with $P_1''\ \mathcal{R}_1\mathcal{R}_2\ P_2''$. Hence $\mathcal{R}_1\mathcal{R}_2$ is also a coarse bisimulation. We conclude that $\sim_c$ is transitive.

□

Finally, we define strong equivalence for weak processes:

Definition 6.6 A binary relation $\mathcal{R}$ on weak processes in an ATS is called a *strong bisimulation* if for all $P, Q \in \mathcal{W}$, $P\ \mathcal{R}\ Q$ implies:

i. For every $p \in Int(P)$, there exists $q \in Int(Q)$ such that for all $a \in Act$ and for all $P' \in \mathcal{W}$: $p{=}a{\Rightarrow}P'$ implies for some $Q' \in \mathcal{W}$, $q{=}a{\Rightarrow}Q'$ and $P'\ \mathcal{R}\ Q'$.

ii. For every $q \in Int(Q)$, there exists $p \in Int(P)$ such that for all $a \in Act$ and for all $Q' \in \mathcal{W}$: $q{=}a{\Rightarrow}Q'$ implies for some $P' \in \mathcal{W}$, $p{=}a{\Rightarrow}P'$ and $P'\ \mathcal{R}\ Q'$.

iii. P has a terminal internal state iff Q has a terminal internal state.

Two weak processes P and Q are *strongly equivalent*, written $P \simeq Q$, if there exists a strong bisimulation containing the pair $\langle P, Q\rangle$.

Proposition 6.7 $\simeq$ *is an equivalence on the weak processes of an ATS.*

Proof We proceed as we did for coarse bisimulation. We show that $\simeq$ is (1) reflexive, (2) symmetric, and (3) transitive.

1. *Reflexivity*: It can easily be verified that the identity relation on $\mathcal{W}$ given by $id_{\mathcal{W}} \overset{\text{def}}{=} \{\langle P, P\rangle : P \in \mathcal{W}\}$ is a strong bisimulation. Thus $\simeq$ is reflexive.

2. *Symmetry*: Let $\mathcal{R}$ be a strong bisimulation on $\mathcal{W}$. By the symmetry of Definition 6.6, $\mathcal{R}^{-1}$ is also a strong bisimulation. Therefore, $\simeq$ is symmetric.

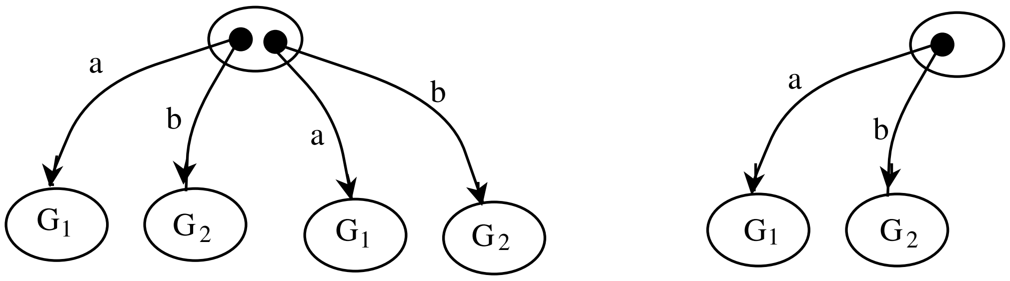


Figure 14: Two weak processes which are equated by strong equivalence.

3. *Transitivity*: Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two strong bisimulations on $\mathcal{W}$. Suppose $P_1\ \mathcal{R}_1\mathcal{R}_2\ P_2$. Then for some $P \in \mathcal{W}$, $P_1\ \mathcal{R}_1\ P$ and $P\ \mathcal{R}_2\ P_2$. Clearly, P_1 has a terminal internal state iff P_2 has a terminal internal state. Now let $P_1[s_1]{=}a{\Rightarrow}P_1'$. Then there exists s such that for some $P' \in \mathcal{W}$, $P[s]{=}a{\Rightarrow}P'$ and $P_1'\ \mathcal{R}_1\mathcal{R}_2\ P'$. Also, since $P\ \mathcal{R}_2\ P_2$, there exists s_2 such that for some $P_2' \in \mathcal{W}$, $P_2[s_2]{=}a{\Rightarrow}P_2'$ with $P'\ \mathcal{R}_2\ P_2'$. Therefore $P_1'\ \mathcal{R}_1\mathcal{R}_2\ P_2'$. By similar reasoning, we can also find, for every $s_2 \in \mathit{Int}(P_2)$ satisfying $s_2{=}a{\Rightarrow}P_2'$, an internal state $s_1 \in \mathit{Int}(P_1)$ and a process $P_1' \in \mathcal{W}$ such that $s_1{=}a{\Rightarrow}P_1'$ and $P_1'\ \mathcal{R}_1\mathcal{R}_2\ P_2'$. Hence, $\mathcal{R}_1\mathcal{R}_2$ is also a strong bisimulation. We conclude that $\simeq$ is transitive.

□

As in the case of strong processes, the type of internal nondeterminism which is not externally observable remains undetected with $\simeq$. For example, the processes shown in Figure 14 are successfully equated by $\simeq$.

6.2 Weak vs. Strong Processes without Divergence

The equivalences introduced in the previous subsection can be summarized as follows:

1. $\sim$ denotes strong equivalence on strong processes; this corresponds to the strongest behavioral equivalence over strong processes that we consider in this paper.

2. $\simeq$ denotes strong equivalence on weak processes; this corresponds to the strongest relevant equivalence over weak processes.

3. $\sim_c$ denotes coarse equivalence on strong processes; it is by this equivalence the weak processes (modulo $\simeq$) of an ATS are characterized in terms of the strong processes (modulo $\sim$) of a related LTS.

Point (3) above is yet to be proved and is the subject of this subsection. We begin by identifying a specific subset of strong processes which we qualify by the adjective *quasi-deterministic*. In the following, we assume that the set of external actions, Act, is fixed.

Definition 6.8 Let $\langle \mathcal{S}, Act, -\cdot\rightarrow \rangle$ be a LTS. The set of *quasi-deterministic* processes, $\mathcal{S}_q$, is the largest set contained in $\mathcal{S}$ which satisfies for all $Q \in \mathcal{S}_q$:

i. $Q{-}a{\rightarrow}Q'$ and $Q{-}a{\rightarrow}Q''$ implies $Q' = Q''$.

ii. $Q{-}a{\rightarrow}Q'$ implies Q' is quasi-deterministic.

A strong process $P \in \mathcal{S}$ is called quasi-deterministic if it belongs to $\mathcal{S}_q$. A LTS is called quasi-deterministic is $\mathcal{S} = \mathcal{S}_q$.

Note that in the above definition, we do not forbid multiple internal transitions, hence the term quasi-deterministic.

We proceed by defining two transformations: γ and ξ. The transformation γ maps a quasi-deterministic LTS into a corresponding ATS, and ξ does the opposite. These transformations are specified by means of inference rules using the following syntax:

$$\mathit{rule\ name}\frac{\mathit{premises}}{\mathit{conclusions}}\ (\mathit{extra\ conditions})$$

First we define γ. Let $\mathbf{L} = \langle \mathcal{S}_q, Act, -\cdot\rightarrow \rangle$ be a quasi-deterministic LTS.

Definition 6.9 $\gamma(\mathbf{L}) \stackrel{\text{def}}{=} \langle \gamma(\mathcal{S}_q), \mathcal{I}, Act, Int, =\cdot\Rightarrow \rangle$, where

i. $\gamma(\mathcal{S}_q) \stackrel{\text{def}}{=} \{\gamma P : P \in \mathcal{S}_q\}$,

ii. $\mathcal{I} \stackrel{\text{def}}{=} \{s_P : P \in \mathcal{S}_q\} \cup \{s_{PQ} : P, Q \in \mathcal{S}_q\}$,

iii. $=\cdot\Rightarrow$ is the smallest relation and Int is such that the $Int(\gamma P)$, where $\gamma P \in \gamma(\mathcal{S}_q)$, are the smallest sets which satisfy the following inference rules:

$$\gamma 1\ \frac{P \text{ is } \mathit{terminal}}{\gamma P[s_P]} \qquad \gamma 2\ \frac{P \text{ is } \mathit{stable},\quad P{-}a{\rightarrow}Q}{\gamma P[s_P]{=}a{\Rightarrow}\gamma Q}\ (a \in Act)$$

$$\gamma 3\ \frac{P \text{ is } \mathit{unstable},\quad P{-}\tau^n{\rightarrow}P'{-}a{\rightarrow}Q}{\gamma P[s_{PP'}]{=}a{\Rightarrow}\gamma Q}\ (a \in Act, n \in Nat)$$

Now let $\mathbf{A} = \langle \mathcal{W}, \mathcal{I}, Act, Int, =\cdot\Rightarrow \rangle$ be an ATS. The transformation ξ, which constructs a quasi-deterministic LTS from a given ATS, is defined as follows:

Definition 6.10 $\xi(\mathbf{A}) \stackrel{\text{def}}{=} \langle \xi(\mathcal{W}), Act, -\cdot\rightarrow \rangle$, where

i. $\xi(\mathcal{W}) \stackrel{\text{def}}{=} \{\xi P : P \in \mathcal{W}\} \cup \{\xi s : s \in \mathcal{I}\}$,

ii. $-\!\cdot\!\!\rightarrow$ is the smallest relation which satisfies the following inference rules:

$$\xi 1 \ \frac{Int(P) = \emptyset}{\xi P {-\tau\rightarrow} \xi P} \qquad \xi 2 \ \frac{P \text{ is } \textit{stable}, \quad P[s]{=}a{\Rightarrow}Q}{\xi P {-a\rightarrow} \xi Q}$$

$$\xi 3 \ \frac{P \text{ is } \textit{unstable}, \quad P[s]{=}a{\Rightarrow}Q}{\xi P {-\tau\rightarrow} \xi s {-a\rightarrow} \xi Q}$$

The following properties of γ and ξ can be inferred from the relevant definitions. Let P be a quasi-deterministic strong process and Q be a weak process. We have:

1. The premises of the rules $\gamma 0$ to $\gamma 3$ are mutually exclusive.
2. The premises of the rules $\xi 1$ to $\xi 3$ are mutually exclusive.
3. P is terminal iff γP is terminal.
4. Q is terminal iff ξQ is terminal.
5. P is stable iff γP is stable.
6. Q is stable iff ξP is stable.

Definition 6.11 Let $\mathbf{L}_1 = \langle \mathcal{S}_1, Act, -\!\cdot\!\!\rightarrow_1 \rangle$ and $\mathbf{L}_2 = \langle \mathcal{S}_2, Act, -\!\cdot\!\!\rightarrow_2 \rangle$ be two LTSs such that $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$. Similarly, let $\mathbf{A}_1 = \langle \mathcal{W}_1, \mathcal{I}_1, Act, Int_1, =\!\cdot\!\!\Rightarrow_1 \rangle$ and $\mathbf{A}_2 = \langle \mathcal{W}_2, \mathcal{I}_2, Act, Int_2, =\!\cdot\!\!\Rightarrow_2 \rangle$ be two ATSs such that $\mathcal{W}_1 \cap \mathcal{W}_2 = \emptyset$ and $\mathcal{I}_1 \cap \mathcal{I}_2 = \emptyset$. Then

i. $\mathbf{S_1} \cup \mathbf{S_2} \stackrel{\text{def}}{=} \langle \mathcal{S}_1 \cup \mathcal{S}_2, Act, -\!\cdot\!\!\rightarrow_1 \cup -\!\cdot\!\!\rightarrow_2 \rangle$.

ii. $\mathbf{W_1} \cup \mathbf{W_2} \stackrel{\text{def}}{=} \langle \mathcal{W}_1 \cup \mathcal{W}_2, \mathcal{I}_1 \cup \mathcal{I}_2, Act, Int_1 \cup Int_2, =\!\cdot\!\!\Rightarrow_1 \cup =\!\cdot\!\!\Rightarrow_2 \rangle$.

We now show that γ and ξ are equivalence preserving homomorphisms which are inversely related to each other:

Theorem 6.12

i. *γ is a homomorphism from $\langle \mathcal{S}_q, \sim_c \rangle$ to $\langle \gamma(\mathcal{S}_q), \simeq \rangle$.*

ii. *For every $Q \in \mathcal{S}_q$ in a quasi-deterministic LTS $\mathbf{L}$, we have $\xi\gamma Q \sim_c Q$ in the LTS $\mathbf{L} \cup \xi(\gamma(\mathbf{L}))$.*

iii. *ξ is a homomorphism from $\langle \mathcal{W}, \simeq \rangle$ to $\langle \xi(\mathcal{W}), \sim \rangle$.*

iv. For every $W \in \mathcal{W}$ in an ATS $\mathbf{A}$, we have $\gamma\xi W \simeq W$ in the ATS $\mathbf{A} \cup \gamma(\xi(\mathbf{A}))$.

Proof We show parts (i), (ii), and (iii). The proof of part (iv) is similar to that of part (ii).

For part (i), we have to show that for every $P, Q \in \mathcal{S}_q$, $P \sim_c Q$ implies $\gamma P \simeq \gamma Q$. So let $P \sim_c Q$. Then there exists a coarse bisimulation $\mathcal{R}_c$ containing the pair $\langle P, Q\rangle$. One can then define a strong bisimulation $\mathcal{R}_{ws}$ as the smallest relation which satisfies the rule:

$$\mathbf{Rws}\frac{\langle P', Q'\rangle \in \mathcal{R}_c}{\langle \gamma P', \gamma Q'\rangle \in \mathcal{R}_{ws}}$$

It is easy to verify, by using the rules $\gamma1$ to $\gamma3$, that $\mathcal{R}_{ws}$ constructed in this way indeed yields a strong bisimulation over $\gamma(\mathcal{S}_q)$. Since $\langle \gamma P, \gamma Q\rangle \in \mathcal{R}_{ws}$, we can conclude that $\gamma P \simeq \gamma Q$.

To prove part (ii), we construct a relation $\mathcal{R}_c$ containing the pair $\langle P, \xi\gamma P\rangle$, and then show that $\mathcal{R}_c$ is a coarse bisimulation. We define $\mathcal{R}_c$ as the smallest relation satisfying the following rules:

$$\mathbf{Rc1}\frac{}{\langle P, \xi\gamma P\rangle \in \mathcal{R}_c} \quad (P \in \mathcal{S}_q)$$

$$\mathbf{Rc2}\frac{\langle R, \xi\gamma R\rangle \in \mathcal{R}_c, \quad R\text{--}\tau^n a\text{→}Q}{\langle Q, \xi\gamma Q\rangle \in \mathcal{R}_c} \quad (a \in Act,\ n \in Nat)$$

First, observe that the premises of the rules $\gamma1$ to $\gamma3$ are mutually exclusive. Similarly for the rules $\xi1$ to $\xi3$.

Clause (iii) of Definition 6.3 can easily be seen to hold true for any pair $\langle R, \xi\gamma R\rangle \in \mathcal{R}_c$. The clauses (i) and (ii) remain to be checked:

Let $R' \in ID(R)$ such that $R'\text{--}a\text{→}Q$ for $a \in Act$. Then by **Rc2** above, $\langle Q, \xi\gamma Q\rangle \in \mathcal{R}_c$. On the one hand, if R is stable then by $\gamma2$, $\gamma R[r_R]{=}a{\Rightarrow}\gamma Q$. Since r_R is γR's only internal state, γR is stable as well. Subsequently, by applying $\xi2$, we obtain $\xi\gamma R\text{--}a\text{→}\xi\gamma Q$. Since $\xi\gamma R \in ID(\xi\gamma R)$, $\xi\gamma R$ itself is the sought internal derivative of $\xi\gamma R$ which satisfies clause (i) of Definition 6.3 when R is stable. On the other hand, if R is unstable then by $\gamma3$, $\gamma R[r_{RR'}]{=}a{\Rightarrow}\gamma Q$. Since R is unstable, γR must also be unstable. Then from $\xi3$, we obtain $\xi\gamma R\text{--}\tau\text{→}\xi r_{RR'}\text{--}a\text{→}\xi\gamma Q$. Since $\xi r_{RR'} \in ID(\xi\gamma R)$, $\xi r_{RR'}$ is the sought internal derivative of $\xi\gamma R$ which satisfies clause (i) of Definition 6.3 when R is unstable.

Now let $\xi W \in ID(\xi\gamma R)$ such that $\xi W\text{--}a\text{→}\xi\gamma Q$ for $a \in Act$. On the one hand, if $\xi\gamma R$ is stable then $W = \gamma R$ and γR must be stable as well. Since the transition $\xi\gamma R\text{--}a\text{→}\xi\gamma Q$ could have been inferred only by applying rule $\xi2$, the corresponding premise of $\xi2$ must hold true: $\gamma R[r_R]{=}a{\Rightarrow}\gamma Q$. By similar reasoning, from rule $\gamma2$, we obtain $R\text{--}a\text{→}Q$. Then by **Rc2** above, $\langle Q, \xi\gamma Q\rangle \in \mathcal{R}_c$. Since $R \in ID(R)$, R itself is the sought internal derivative of R which satisfies clause (ii) of Definition 6.3 when $\xi\gamma R$ is stable. On the other hand, if $\xi\gamma R$ is unstable, we must have $W = r_{RR'}$ for some $R' \in ID(R)$ such that $\xi\gamma R{-}\tau{\rightarrow}\xi r_{RR'}{-}a{\rightarrow}\xi\gamma Q$. Since $\xi\gamma R$ is unstable, γR must also be unstable. The only rule from which this sequence of transitions can be inferred is $\xi 3$. Therefore, the corresponding premise of $\xi 3$ must hold true: $\gamma R[r_{RR'}]{=}a{\Rightarrow}\gamma Q$. Subsequently, with similar reasoning, we can use the rule $\gamma 3$ to produce the premise from which $\gamma R[r_{RR'}]{=}a{\Rightarrow}\gamma Q$ was obtained: $R{-}\tau^n{\rightarrow}R'{-}a{\rightarrow}Q$ for some $n \in Nat$. Thus $R' \in ID(R)$, and by **Rc2** above, $\langle Q, \xi\gamma Q\rangle \in \mathcal{R}_c$. Consequently, R' is the sought internal derivative of R which satisfies clause (ii) of Definition 6.3 when $\xi\gamma R$ is unstable.

We conclude that $\mathcal{R}_c$ is a coarse bisimulation.

For part (iii), we have to show that for every $W, Z \in \mathcal{W}$, $W \simeq Z$ implies $\xi W \sim \xi Z$. Let $W \simeq Z$. Then there exists a strong bisimulation $\mathcal{R}_{ws}$ containing the pair $\langle W, Z\rangle$. We proceed the same way as in part (i), except that this time we construct a strong bisimulation $\mathcal{R}_s$ containing the pair $\langle \xi W, \xi Z\rangle$ using $\mathcal{R}_{ws}$. $\mathcal{R}_s$ is defined as the smallest relation satisfying the following rules:

$$\textbf{Rs1}\frac{\langle W', Z'\rangle \in \mathcal{R}_{ws}}{\langle \xi W', \xi Z'\rangle \in \mathcal{R}_s}$$

$$\textbf{Rs2}\frac{\langle W', Z'\rangle \in \mathcal{R}_{ws}, \quad W'[w]{=}a{\Rightarrow}W'', \quad Z'[z]{=}a{\Rightarrow}Z''}{\langle \xi w, \xi z\rangle \in \mathcal{R}_s}$$

It is easy to verify, by using rules $\xi 1$ to $\xi 3$, that $\mathcal{R}_s$ is indeed a strong bisimulation over $\xi(\mathcal{W})$.

For part (iv) of the theorem, we proceed the same way as in part (iii), except that this time we construct a strong bisimulation $\mathcal{R}_{ws}$ on weak processes. $\mathcal{R}_{ws}$ is defined as the smallest relation satisfying the following rules:

$$\textbf{Rws1}\frac{}{\langle W, \gamma\xi W\rangle \in \mathcal{R}_{ws}} \ (W \in \mathcal{W})$$

$$\textbf{Rws2}\frac{\langle Z, \gamma\xi Z\rangle \in \mathcal{R}_{ws}, \quad Z[z]{=}a{\Rightarrow}V}{\langle V, \gamma\xi V\rangle \in \mathcal{R}_{ws}}$$

□

Notation 6.13 Let A be an arbitrary set and $\approx$ an equivalence relation on A. Then $A/{\approx}$ denotes the set of all *equivalence classes* induced by $\approx$ on A.

Corollary 6.14 *There exists a one-to-one mapping between* $(\mathcal{S}_q \cup \xi(\mathcal{W}))/{\sim_c}$ *and* $(\mathcal{W} \cup \gamma(\xi(\mathcal{W})) \cup \gamma(\mathcal{S}_q))/{\simeq}$.

Proof Follows from Theorem 6.12. □

Definition 6.15 For two LTSs $\mathbf{L}_1$ and $\mathbf{L}_2$ sharing a common set of actions, define $\mathbf{L}_1 \subseteq \mathbf{L}_2$ if $\mathcal{S}_1 \subseteq \mathcal{S}_2$ and $-\!\cdot\!\!\rightarrow_1 \subseteq -\!\cdot\!\!\rightarrow_2$. Similarly, define for two ATSs $\mathbf{A}_1$ and $\mathbf{A}_2$ sharing a common set of actions, $\mathbf{A}_1 \subseteq \mathbf{A}_2$ if $\mathcal{W}_1 \subseteq \mathcal{W}_2$, $\mathcal{I}_1 \subseteq \mathcal{I}_2$, $Int_1 \subseteq Int_2$, and $=\!\cdot\!\!\Rightarrow_1 \subseteq =\!\cdot\!\!\Rightarrow_2$.

Lemma 6.16 *For every pair $\langle \mathbf{L}', \mathbf{A}' \rangle$ where $\mathbf{L}'$ is a non-trivial quasi-deterministic LTS and $\mathbf{A}'$ is a non-trivial ATS, there exist a pair $\langle \mathbf{L}, \mathbf{A} \rangle$ with $\mathbf{L}' \subseteq \mathbf{L}$, $\mathbf{A}' \subseteq \mathbf{A}$, $\mathcal{S}$ denoting the strong processes of $\mathbf{L}$, and $\mathcal{W}$ denoting the weak processes of $\mathbf{A}$, such that every well-defined function from $\mathcal{S}/\!\sim$ onto $\mathcal{W}/\!\simeq$ is strictly many-to-one.*

Proof Let $e^{\mathbf{L}}_{\mathcal{R}}(P)$ denote the equivalence class in $\mathcal{S}/\mathcal{R}$ of the strong process P in $\mathbf{L}$ with $\mathcal{S}$ being the the set of strong processes of $\mathbf{L}$.

First find a LTS $\mathbf{L}''$ such that $\mathbf{L}' \subseteq \mathbf{L}''$ and $e^{\mathbf{L}}_{\sim}(P'') \subset e^{\mathbf{L}''}_{\sim_c}(P'')$ for some $P'' \in \mathcal{S}''$, where $\mathcal{S}''$ is the set of strong processes of $\mathbf{L}''$. Such an LTS exists (and is easy to construct) because $\sim_c$ is in strictly weaker than $\sim$. One can then choose the pair $\langle \mathbf{L}, \mathbf{A} \rangle$ such that $\mathbf{L} = \mathbf{L}'' \cup \xi(\mathbf{A}')$ and $\mathbf{A} = \mathbf{A}' \cup \gamma(\xi(\mathbf{A}')) \cup \gamma(\mathbf{L}'')$. The result is then immediate from Corollary 6.14 and Proposition 6.4. $\square$

Lemma 6.16 states that, in general, all surjective mappings from the strong processes to the weak processes must be many-to-one, up to strong equivalences. *This in turn suggests that weak processes are less expressive than strong processes under the strongest perspective of external behavior that we imagine for both models.*

This fact can also be expressed from a second, perhaps more illuminating, perspective. We fix the transformation ξ; since unlike γ, ξ preserves strong equivalence across models. Subsequently, we show that ξ does not have a strong equivalence preserving inverse.

Lemma 6.17 *There does not exist a strong equivalence preserving transformation γ' which satisfies both $P \sim \xi\gamma' P$ for every strong process P and $W \simeq \gamma'\xi W$ for every weak process W.*

Proof The proof is by contradiction. Let us suppose that such a transformation exists. If γ' is strong equivalence preserving then by definition, for every pair of quasi-deterministic processes P and Q, $P \sim Q$ implies $\gamma' P \simeq \gamma' Q$.

Now consider a quasi-deterministic process R such that $R \not\sim \xi\gamma R$. Such a strong process exists because γ does not in general satisfy $P \sim \xi\gamma P$, for every P (the process on the left hand side of Figure 8 is an example.) But by the hypothesis, we must still have $R \sim \xi\gamma' R$. We show that $R \not\sim \xi\gamma R$ and $R \sim \xi\gamma' R$ cannot be satisfied simultaneously.

If $\gamma' R \simeq \gamma R$ then we must have, by part (*iii*) of Theorem 6.12, $\xi\gamma' R \sim \xi\gamma R$. But since $R \sim \xi\gamma'$, we obtain $R \sim \xi\gamma R$. This in turn contradicts with the hypothesis that $R \not\sim \xi\gamma R$.

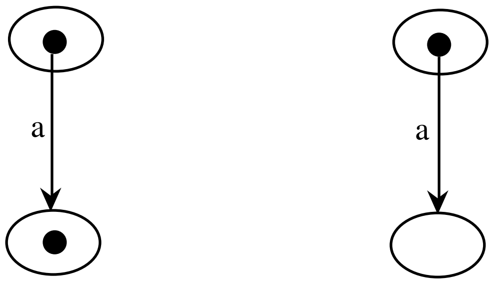


Figure 15: Two weak processes which can be distinguished by strong equivalence.

Now suppose $\gamma' R \not\simeq \gamma R$. On the one hand, since $\sim_c$ is weaker than $\sim$, $R \sim \xi\gamma' R$ implies $R \sim_c \xi\gamma' R$. On the other hand, by part (*ii*) of Theorem 6.12, $R \sim_c \xi\gamma R$. Therefore, we must have $\xi\gamma R \sim_c \xi\gamma' R$. Subsequently, by using part (*i*) of Theorem 6.12, we obtain $\gamma\xi\gamma R \simeq \gamma\xi\gamma' R$. Then we can use part (*iv*) on both sides of this latter derivation to obtain $\gamma R \simeq \gamma' R$, leading to a contradiction with the premise of this paragraph. $\square$

7 Divergence

7.1 Weak Processes with Divergence

It is sometimes important to be able to represent a system's potential to undertake an infinite internal computation. In the literature, this phenomenon — which is sometimes associated with "undefinedness" — is referred to as *divergence*. In LTSs, divergence coincides with the ability of a strong process to perform an infinite sequence of τ actions.

The basic ATS model provides an explicit representation for a system which has no internal states, thus no external behavior. For instance, the two weak processes depicted in Figure 15 can be distinguished from one another. Here the process on the left represents a system which successfully terminates upon performing the action a, whereas the process on the right represents one which "hangs" (or internally computes forever) upon performing the same action. However, this distinction is superficial without the ability to represent as a distinct object a process which may either perform external actions or diverge.

To treat divergence on a finer scale, we associate with each weak process a *divergence property set*, or *divergence set* in short. When interpreted properly, this set determines whether or not the weak process in question may diverge, and when it does, the type of divergence it exhibits. Divergence properties and drawn from the set of natural numbers *Nat*. If a given process P satisfies the divergence property

$n \in \mathit{Nat}$, then n is included in its divergence set $\mathit{Div}(P)$. In this view, the basic ATS model is extended as follows:

Definition 7.1 An *Extended Abstract Transition System* (EATS) is a structure

$$\langle \mathcal{W}, \mathcal{I}, \mathit{Act}, \mathit{Int}, \mathit{Div}, =\cdot\Rightarrow \rangle$$

where $\langle \mathcal{W}, \mathcal{I}, \mathit{Act}, \mathit{Int}, =\cdot\Rightarrow \rangle$ is an ATS and Div is a function from $\mathcal{W}$ to $\wp(\mathit{Nat})$ satisfying for all $P \in \mathcal{W}$:

i. If $\mathit{Int}(P) = \emptyset$ then $\mathit{Div}(P) = \{0\}$.

ii. For all $n, m \in \mathit{Nat}$ such that $m \leq n$, $n \in \mathit{Div}(P)$ implies $m \in \mathit{Div}(P)$.

Note that we allow a divergence set to be empty — precisely, when the owner weak process is convergent. The divergence property i indicates an infinite internal computation along which intermediate processes with at least one internal state recur at least i times, but not infinitely often, unless the process which is the source of the infinite computation enjoys the divergence property j for every $j \in \mathit{Nat}$.

Notation 7.2 For convenience, we write nP, read as "P has divergence property n", to abbreviate $n \in \mathit{Div}(P)$. Conversely, we write $\neg nP$ to abbreviate $n \notin \mathit{Div}(P)$.

Divergence typically arises when some of the external actions of a system are hidden. For example, it may occur when two systems are interconnected and encapsulated in a black box, from the outside of which the interactions between the two systems are invisible.

As an example, consider the weak processes P_0 and Q_0 depicted in Figure 16. Suppose the action a is hidden in both processes, so that the execution of a is associated with the internal behavior of P_0 and Q_0. Recall that $P\,\mathbf{hide}\,\{a\}$ represents the behavior of P after the action a has been hidden. Since both P_0 and Q_0 may perform an infinite sequence of a-transitions, the processes $P_0\,\mathbf{hide}\,\{a\}$ and $Q_0\,\mathbf{hide}\,\{a\}$ may diverge. However, the types of divergence that the two processes exhibit are different. On the one hand, $Q_0\,\mathbf{hide}\,\{a\}$ has a pathological behavior; it may diverge never to offer an internal state with an external behavior — i.e., one which either is terminal or offers an external action. On the other hand, the behavior of $P_0\,\mathbf{hide}\,\{a\}$ is not pathological and usually acceptable: it may diverge, but while it diverges, the potential of an intermediate internal state with an external behavior is never ruled out. In the latter case, divergence is in the sense of an internal loop during which the action b is offered infinitely often. The type of divergence that $P_0\,\mathbf{hide}\,\{a\}$ exhibits is therefore weaker; we refer to it as *weak divergence*. That of $Q_0\,\mathbf{hide}\{a\}$ is the stronger and the pathological version; we refer to it as *strong divergence*. It is not possible to

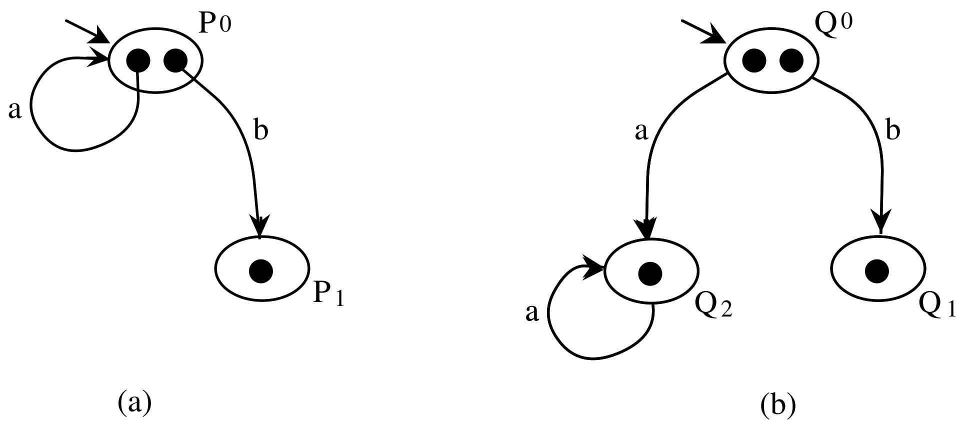


Figure 16: Two weak processes which exhibit different kinds of divergence upon the hiding of the action a.

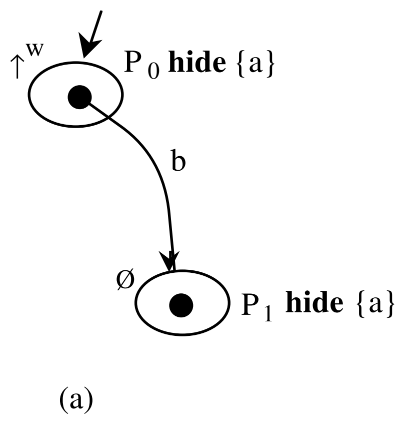
recover from strong divergence beyond a certain point, whereas in weak divergence recovery always remains a possibility. The divergent weak processes $P_0\,\mathbf{hide}\,\{a\}$ and $Q_0\,\mathbf{hide}\,\{a\}$ are depicted in Figure 17. Note that the two transition graphs differ only in terms of the divergence sets of the nodes P_0 and Q_0.

Definition 7.3 Let $P \in \mathcal{W}$ be a weak process in an EATS.

1. P is called *divergent*, written $\uparrow P$, if $\mathit{Div}(P) \neq \emptyset$.
2. P is called *convergent*, written $\downarrow P$, if $\mathit{Div}(P) = \emptyset$.
3. P is called *strongly divergent*, written $\uparrow^s P$, if $\max(\mathit{Div}(P)) = i$ for some $i \in \mathit{Nat}$. Therefore, $\uparrow^s P$ if for some $i \in \mathit{Nat}$, $\neg iP$.
4. P is *weakly divergent*, written $\uparrow^w P$, if $\mathit{Div}(P) = \mathit{Nat}$. Therefore, $\uparrow^w P$ if we have iP for every $i \in \mathit{Nat}$.
5. For $i \in \mathit{Nat}$, P is called *i-divergent*, written $\uparrow^i P$, if $\max(\mathit{Div}(P)) = i$. Therefore, $\uparrow^i P$ if we have iP and $\neg(i+1)P$.

Weak divergence can be interpreted as the *limit* of strong divergence by defining a preorder on weak processes based upon divergence properties:

Definition 7.4 Let P, Q be strongly equivalent weak processes in an EATS. Write $P \leq_\downarrow Q$ if both $\downarrow P$ implies $\downarrow Q$ and $\uparrow P$ implies $\mathit{Div}(P) \subseteq \mathit{Div}(Q)$.



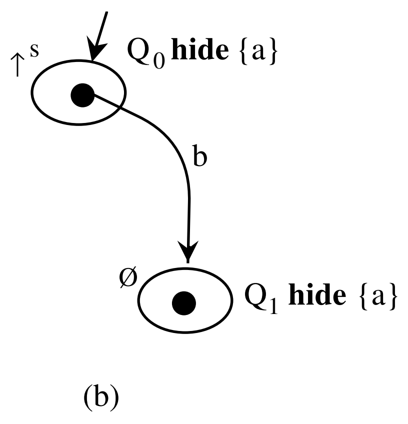


Figure 17: Processes P_0 and Q_0 of Figure 16 after the action a has been hidden. The label $\uparrow^w$ indicates weak divergence, whereas $\uparrow^s$ indicates strong divergence.

The preorder $\leq_\downarrow$ induces a complete lattice:

$$\begin{array}{c} \downarrow = \top \\ | \\ \uparrow^w \\ \vdots \\ \uparrow^2 \\ | \\ \uparrow^1 \\ | \\ \uparrow^0 = \bot \end{array}$$

Consider an infinite chain of strongly equivalent and strongly divergent weak processes $P_0, P_1, P_2, \ldots$ which satisfy $P_i \leq_\downarrow P_{i+1}$ with $Div(P_i) \subset Div(P_{i+1})$ for all $i \in Nat$. The limit — or the least upper bound with respect to the partial order induced by $\leq_\downarrow$ — of this chain is a weakly divergent process which is strongly equivalent to each of the P_i. This interpretation gives rise to a new perspective so far as the relationship between weak and strong divergence is concerned: On the one hand, convergence and strong divergence are deemed elementary properties, strong divergence being "less defined" a property than convergence. In other words, these are algebraically *finite* properties, typically of non-recursive processes. On the other hand, we have weak divergence, which falls between convergence and strong divergence. Just as infinite processes can be derived from their finite approximations within a complete partial order structure [14, 10], weak divergence can be derived from its own finite counterpart, strong divergence. That is, the limit of an infinite chain of strongly divergent processes which are increasingly more convergent yields a weakly divergent process. This limit cannot result in a convergent process since divergence always remains possible, even in the limit. Thus the limit is the closest a divergent process can get to a convergent one.

Weak divergence is typically exhibited by recursive processes — up to hiding, or abstraction of internal behavior. In most cases, we can think of a weakly divergent process as an abstraction of a recursive process whose certain actions have been hidden, causing it to diverge, but not persistently, or in a pathological way.

Note that although it is in practice not critical to distinguish between i- and j-divergence for $i \neq j$, the notion of i-divergence is still useful. By the limit interpretation, if for a given process one can infer i-divergence for all $i \in Nat$, then it can be concluded that the process in question weakly diverges. Therefore, mathematical induction on i-divergence can be used for deciding weak divergence. This idea will be applied in Section 8.2 when we assign a weak process semantics to the hiding construct of our example process algebra.

7.2 Divergences for Strong Processes

The different forms of divergence which were initially defined for weak processes may be interpreted over strong processes in a straightforward manner. A strong process diverges if it may perform an uninterrupted infinite sequence of internal transitions. In strong divergence, processes with at least one internal derivative are only finitely recurrent, whereas in weak divergence, processes with at least one internal derivative are infinitely recurrent.

Definition 7.5 Given a LTS, let Θ be an indexing function over $\mathcal{S}$. If for every $i \in Nat$, $\Theta_i {-}\tau{\rightarrow} \Theta_{i+1}$ then Θ is called an *infinite internal computation.* The strong process Θ_0 is called the *source* of Θ. The set of processes along Θ (i.e., in the range of Θ) with at least one internal derivative is given by

$$\Delta(\Theta) \overset{\text{def}}{=} \{i : ID(\Theta_i) \neq \emptyset\}$$

Definition 7.6 Let $P \in \mathcal{S}$ be a strong process in a LTS.

1. P is *divergent*, written $P{\uparrow}$, if there exists an infinite internal computation whose source is P.

2. P is called *convergent*, written $P{\downarrow}$, if there does *not* exist an infinite internal computation whose source is P.

3. P is *strongly divergent*, written $P{\uparrow}^s$, if it is divergent and for each infinite internal computation Θ with $P = \Theta_0$, $\Delta(\Theta)$ is *finite.*

4. P is *weakly divergent*, written $P{\uparrow}^w$, if it is divergent and for each infinite internal computation Θ with $P = \Theta_0$, $\Delta(\Theta)$ is *infinite*.

5. For $i \in Nat$, P is called *i-divergent*, written $P{\uparrow}^i$, if $P{\uparrow}^s$ and $i = \min\{\Delta(\Theta) :$ Θ is an infinite internal computation with $\Theta_0 = P\}$

Let τ^∞ be that indexing function over $Act \cup \{\tau\}$ defined by $\tau_i^\infty = \tau$ for all $i \in Nat$. Thus τ^∞ represents an infinite sequence of τ actions. It is obvious from the above definitions that $P{\uparrow}$ whenever $P{-}\tau^\infty{\rightarrow}$; otherwise, $P{\downarrow}$.

7.3 Divergence Discriminating Equivalences

We can easily extend the two bisimulation equivalences — $\simeq$ on strong processes and $\sim_c$ on weak processes — to give them the level of discriminating power necessary to distinguish among convergence, strong divergence, and weak divergence.

Definition 7.7 A binary relation $\mathcal{R}$ on strong processes in a LTS is called a *divergence discriminating coarse bisimulation* if $\mathcal{R}$ is a coarse bisimulation and for every $P, Q \in \mathcal{S}$, $P \mathrel{\mathcal{R}} Q$ implies:

i. $P{\downarrow}$ iff $Q{\downarrow}$.

ii. $P{\uparrow}^s$ iff $Q{\uparrow}^s$.

We write $P \sim_c^{\uparrow} Q$ if there exists a divergence discriminating coarse bisimulation containing the pair $\langle P, Q \rangle$.

Definition 7.8 A binary relation $\mathcal{R}$ on weak processes in an EATS is called a *divergence discriminating strong bisimulation* if $\mathcal{R}$ is a strong bisimulation and for every $P, Q \in \mathcal{W}$, $P \mathrel{\mathcal{R}} Q$ implies:

i. ${\downarrow}P$ iff ${\downarrow}Q$.

ii. ${\uparrow}^s P$ iff $P{\uparrow}^s Q$.

We write $P \simeq^{\uparrow} Q$ if there exists a divergence discriminating strong bisimulation containing the pair $\langle P, Q \rangle$.

As before, both $\sim_c^{\uparrow}$ and $\simeq^{\uparrow}$ are equivalences — the former on the strong processes of a LTS while the latter on the weak processes of an EATS. Note that $\sim_c^{\uparrow}$ and $\simeq^{\uparrow}$ are stronger than $\sim_c$ and $\simeq$, respectively. Although they discriminate among convergence, strong divergence, and weak divergence, neither $\sim_c^{\uparrow}$ nor $\simeq^{\uparrow}$ discriminates between i- and j- divergence for $i \neq j$. Such fine level of granularity would be unnecessary from a practical point of view.

7.4 Weak vs. Strong Processes with Divergence

To take into account divergence, we extend the two transformations γ and ξ defined in Section 6.2 to γ' and ξ'. First we define γ'. As with γ, we let $\mathbf{L} = \langle \mathcal{S}_q, Act, -\!\cdot\!\rightarrow \rangle$ be a quasi-deterministic LTS.

Definition 7.9 $\gamma'(\mathbf{L}) \overset{\text{def}}{=} \langle \gamma'(\mathcal{S}_q), \mathcal{I}, Act, Int, Div, =\!\cdot\!\Rightarrow \rangle$, where

i. $\gamma'(\mathcal{S}_q) \overset{\text{def}}{=} \gamma(\mathcal{S}_q) = \{\gamma P : P \in \mathcal{S}_q\}$,

ii. $\mathcal{I} \overset{\text{def}}{=} \{s_P : P \in \mathcal{S}_q\} \cup \{s_{PQ} : P, Q \in \mathcal{S}_q\}$,

iii. $=\!\cdot\!\Rightarrow$ is the smallest relation, and *Int* and *Div* are such that the $Int(\gamma P)$ and the $Div(\gamma P)$, for $\gamma P \in \gamma'(\mathcal{S}_q)$, are the smallest sets which, in addition to rules $\gamma 0$ to $\gamma 3$ given in Definition 6.9, satisfy the following inference rules:

$$\gamma' 1 \ \frac{P{\uparrow}^w}{n(\gamma P)} \ (n \in Nat) \qquad \gamma' 2 \ \frac{P{\uparrow}^s}{0(\gamma P)}$$

Now let $\mathbf{E} = \langle \mathcal{W}, \mathcal{I}, Act, Int, Div, =\!\cdot\!\Rightarrow, \rangle$ be an EATS.

Definition 7.10 $\xi'(\mathbf{E}) \overset{\text{def}}{=} \langle \xi'(\mathcal{W}), Act, -\!\cdot\!\rightarrow \rangle$, where for $\Omega \notin \{\xi P : P \in \mathcal{W}\}$

i. $\xi'(\mathcal{W}) \overset{\text{def}}{=} \{\xi P : P \in \mathcal{W}\} \cup \{\xi s : s \in Int(P)\} \cup \{\Omega\}$,

ii. $-\!\cdot\!\rightarrow$ is the smallest relation which, in addition to rules $\xi 0$ to $\xi 3$ given in Definition 6.10, satisfies the following inference rules:

$$\xi' 1 \ \frac{{\uparrow}^w P}{\xi P - \tau \rightarrow \xi P} \qquad \xi' 2 \ \frac{}{\Omega - \tau \rightarrow \Omega} \qquad \xi' 3 \ \frac{{\uparrow}^s P}{\xi P - \tau \rightarrow \Omega}$$

The new transformations are now good for mapping a quasi-deterministic LTS to a related EATS and vice versa. The following properties can be seen to hold true from the relevant definitions:

1. ${\uparrow}^s P$ iff $\mathcal{W}'(P){\uparrow}^s$.
2. ${\uparrow}^w P$ iff $\mathcal{W}'(P){\uparrow}^w$.
3. ${\downarrow} P$ iff $\mathcal{W}'(P){\downarrow}$.
4. $Q{\uparrow}^s$ iff ${\uparrow}^s \xi'(Q)$.
5. $Q{\uparrow}^w$ iff ${\uparrow}^w \xi'(Q)$.

6. $Q{\downarrow}$ iff ${\downarrow}\xi'(Q)$.

Definition 7.11 Let $\mathbf{E}_1$ and $\mathbf{E}_2$ be two EATSs such that $\mathcal{W}_1 \cap \mathcal{W}_2 = \emptyset$ and $\mathcal{I}_1 \cap \mathcal{I}_2 = \emptyset$. Define the EATS $\mathbf{E}_1 \cup \mathbf{E}_2$ as:

$$\mathbf{E}_1 \cup \mathbf{E}_2 \overset{\text{def}}{=} \langle \mathcal{W}_1 \cup \mathcal{W}_2, \mathcal{I}_1 \cup \mathcal{I}_2, Act, Int_1 \cup Int_2, Div_1 \cup Div_2, =\!\cdot\!\Rightarrow_1 \cup =\!\cdot\!\Rightarrow_2 \rangle.$$

The following extends Theorem 6.12 and Corollary 6.14 to the divergence discriminating case. The implications are similar.

Theorem 7.12

i. *γ' is a homomorphism from $\langle \mathcal{S}_q, \sim_c^{\uparrow} \rangle$ to $\langle \gamma'(\mathcal{S}_q), \simeq^{\uparrow} \rangle$.*

ii. *For every $Q \in \mathcal{S}_q$ in a quasi-deterministic LTS $\mathbf{L}$, we have $\xi'\gamma'Q \sim_c^{\uparrow} Q$ in the LTS $\mathbf{L} \cup \xi'(\gamma'(\mathbf{L}))$.*

iii. *ξ' is a homomorphism from $\langle \mathcal{W}, \simeq^{\uparrow} \rangle$ to $\langle \xi'(\mathcal{W}), \sim \rangle$.*

iv. *For every $W \in \mathcal{W}$ in an EATS $\mathbf{E}$, we have $\gamma'\xi'W \simeq^{\uparrow} W$ in the EATS $\mathbf{E} \cup \gamma'(\xi'(\mathbf{E}))$.*

Corollary 7.13 *There exists a one-to-one mapping between $(\mathcal{S}_q \cup \xi'(\mathcal{W}))/\!\sim_c^{\uparrow}$ and $(\mathcal{W} \cup \gamma'(\xi'(\mathcal{W})) \cup \gamma'(\mathcal{S}_q))/\!\simeq^{\uparrow}$.*

8 Operational Semantics

8.1 MPA: A Minimal Process Algebra without τ

In Section 2, we suggested various syntactic constructs for describing concurrent nondeterministic processes. Namely, these are: *NIL*, $a\cdot$, $+$, $\oplus$, $\langle M \rangle$, **hide**M, and **def**. To the above, we now add a new construct: Ω. This set gives rise to a compact process algebra which we refer to as MPA — short for *a Minimal Process Algebra.* MPA brings together the mechanisms needed to express the basic notions of sequentiality, nondeterminism, communication, concurrency, abstraction of internal behavior, and recursion. MPA's constructs are borrowed from other similar languages. For example, Ω, *NIL*, $a\cdot$, $+$, and $\oplus$ are borrowed from Hennessy's example language **EPL** [14], whereas **hide**M and $\langle M \rangle$ are inspired by LOTOS [27].

The syntax of MPA is given by:

$$E ::= \Omega \;\Big|\; NIL \;\Big|\; a\cdot E \;\Big|\; E \,\mathbf{hide}\, M \;\Big|\; E_1 + E_2 \;\Big|\; E_1 \oplus E_2 \;\Big|\; E_1 \langle M \rangle E_2 \;\Big|\; X \,\mathbf{def}\, E$$

where $a \in Act$; $M \subseteq Act$; X is a *process variable*; and E, E_1, and E_2 are MPA expressions. An MPA expression E is *open* if it contains occurrences of some process variable X which is not *bound* by a sub-expression E' of E having the form X **def** E'. Otherwise E is *closed*.

For example, in the open expression

$$a{\cdot}X \oplus a{\cdot}(X\ \mathbf{def}\ a{\cdot}X \oplus b{\cdot}NIL) + (Y\ \mathbf{def}\ b{\cdot}Y),$$

the first occurrence, from left to right, of X is free, whereas its second and third occurrences are bound by the closed sub-expression X **def** $a{\cdot}X \oplus b{\cdot}NIL$. Note that here all occurrences of the variable Y are bound.

We assume that the binary constructs $+$, $\oplus$, and $\langle M\rangle$ all have the same precedence. Similarly, the unary constructs $a\cdot$ and **hide**M have the same precedence, which supersedes that of the binary constructs. For example the expression $a{\cdot}P \oplus Q$ is read as $(a{\cdot}P) \oplus Q$, whereas the expression $P \oplus Q + R$ is ambiguous without parentheses.

8.2 Weak Process Semantics of MPA

Now we supply an operational semantics for MPA in terms of EATSs. Using the method of structured operational semantics [25, 22], the EATS corresponding to a given closed MPA expression E will be specified in an inductive manner in terms of the EATSs corresponding to the constituents of E. As with the transformations γ and ξ, inference rules of the form

$$\textbf{RuleName}\ \frac{\textit{premises}}{\textit{conclusions}}\ (\textit{extra conditions})$$

will be used. The *premises* and *conclusions* are composed of a series of EATS predicates, referred to as *literals* below, which are interpreted conjunctively. There are three kinds of basic (positive) literals that are allowed in the rules. These will be discussed in the next subsection. Basic literals can be negated, universally quantified, or combined using disjunctions to form more complex literals, subject to the restrictions discussed in the next subsection.

The rules will also express how divergence is inferred and inherited. The first rule, **URule**, will insure that the closure condition of divergence sets is satisfied; i.e., that $(n+1)P$ implies nP for every $n \in Nat$. With this rule, the positive literal $0P$ is equivalent to $\uparrow P$, and conversely, the negative literal $\neg 0P$ is equivalent to $\downarrow P$.

8.2.1 Use of Negative Literal in Premises

In the *premises*, we will allow the following three basic forms of *negative literals*: (1) $p \not= a \Rightarrow$, (2) $Int(P) = \emptyset$, and (3) $\downarrow P$. Note that the literal "P has a terminal internal state p" is actually the literal $p{\not=}a{\Rightarrow}$ with the action a universally quantified over Act; hence, it is covered by (1) above. The literal "$\uparrow P$ implies nP" can be written as "$\downarrow P \vee nP$", and its negative component $\downarrow P = \neg 0P$ is therefore covered by (3) above. Similarly, a literal of the form "for every P' in $g(P)$, $\uparrow(f(P'))$ implies $n(f(P'))$" can be viewed as the literal "$\downarrow(f(P')) \vee n(f(P'))$" with P' universally quantified over the set $g(P)$, and its negative component is therefore covered by (3) above as well. These two latter forms appear in the definition of the predicate *inherits* used in the premises of rules **HideP4** and **HideP5** below.

In general, the use of negative literals in the premises may lead to an inconsistent operational semantics. More specifically, the existence of the intended least relations is not guaranteed with inference systems containing negative premises. However, Groote demonstrated in [13] that restricted use of negative premises can be justified. He proved that if a given set of inference rules are *stratifiable* then they can be shown to define a unique (least) LTS. This result can be readily extended to the structure of an EATS: the only difference is that an EATS defines three relations — namely, $=\cdot\Rightarrow$, *Int*, and *Div* — instead of one. Thus in the EATS rules, we can have three kinds of basic positive literals: (1) $P[p]$, (2) $p{=}a{\Rightarrow}Q$, and (3) nP. However, a literal of the second kind will always be combined with a literal of the first kind, yielding a literal of the form $P[p]{=}a{\Rightarrow}Q$. We also consider this latter form as a basic literal.

A *stratification* Σ is a function which maps a literal allowed in the semantic rules to a corresponding *ordinal number*. The stratification corresponding to the negative literals is the same as the one for the positive ones. Such a mapping induces an inference scheme by imposing restrictions on the order in which the semantic rules can be applied.

As the range of the stratification, we consider the increasing sequence of ordinals

$$1 < 2 < \cdots < \omega < \omega +_o 1 < \omega +_o 2 \cdots < \omega +_o \omega < \omega +_o \omega +_o 1 < \cdots$$

where ω is the smallest transfinite ordinal and $+_o$ denotes addition for ordinal numbers. The reader is referred to [26] and [30] for transfinite ordinals and the properties of ordinal numbers. One problem with ordinal arithmetic (as opposed to cardinal arithmetic) is that $+_o$ is not commutative : $\omega +_o 1 \neq \omega$ but $1 +_o \omega = \omega$. We can easily get around this problem by defining a commutative ordinal addition, $+$, as follows:

$$o_1 + o_2 \overset{\text{def}}{=} \max(o_1 +_o o_2, o_2 +_o o_1)$$

where o_1 and o_2 are two ordinals. This guarantees that $\omega + 1 = 1 + \omega = \omega +_o 1 \neq \omega$.

Before specifying a stratification on EATS literals where process names are EATS expressions, we define a function σ which maps a closed MPL expression to an ordinal o. The function σ is defined inductively as follows:

σ1. $\sigma(NIL) = \sigma(\Omega) = \sigma(X) \overset{\text{def}}{=} 1$.

σ2. $\sigma(a{\cdot}P) = \sigma(P\,\textbf{hide}\,M) \overset{\text{def}}{=} \sigma(P) + 1$.

σ3. $\sigma(P \oplus Q) = \sigma(P + Q) = \sigma(P\langle M\rangle Q) \overset{\text{def}}{=} \sigma(P) + \sigma(Q) + 1$.

σ4. $\sigma(X\ \textbf{def}\ E) \overset{\text{def}}{=} \omega$.

For example, $\sigma((X\ \textbf{def}\ E) \oplus (Y\ \textbf{def}\ F)) = \omega + \omega + 1$. Now we can define the required stratification Σ on the basic positive literals:

Σ1. $\Sigma(P[p]) = \Sigma(P[p]{=}a{\Rightarrow}Q) = \Sigma(0P) \overset{\text{def}}{=} \sigma(P)$.

Σ2. $\Sigma((n+1)P) \overset{\text{def}}{=} \Sigma(nP) + 1$.

For example, since

$$\sigma((X\ \textbf{def}\ E) \oplus (Y\ \textbf{def}\ F)) = \omega + \omega + 1 < \sigma(X\ \textbf{def}\ E) = \omega$$

the behavior of X **def** E should be inferred before that of the super expression $(X\ \textbf{def}\ E) \oplus (Y\ \textbf{def}\ F)$. As far as divergence properties are concerned, (Σ2) above suggests that for any weak process P, $(i+1)$-divergence should be inferred only after i-divergence has been inferred for P.

Let π, π', π'' denote positive literals. The above stratification suggests the following inference scheme: First positive literals π with $\Sigma(\pi) = 1$ are inferred. These can only be inferred using those rules which contain no negative premises. One can then determine which negative literals $\neg\pi$ with $\Sigma(\pi) = 1$ hold true. Subsequently, we can use this information to infer all positive literals π' with $\Sigma(\pi') = 2$, and also some positive literals with $\Sigma(\pi') > 2$. In general, once all the positive literals with an Σ-value o have been inferred, one can determine which negative literals $\neg\pi'$ with $\Sigma(\pi') = o$ hold true, and again using this information, one can determine which positive literals π'' with $\Sigma(\pi'') = o + 1$ hold true, and possibly, the truth value of some of the positive literals π'' with $\Sigma(\pi'') > o + 1$. Note that such a scheme justifies rules of the form

$$\textbf{Form1}\quad \frac{\bigwedge_{k\in K} \pi_k,\ \bigwedge_{\ell\in L} \neg\pi_\ell}{\pi}$$

if $\Sigma(\pi_k) \leq \Sigma(\pi)$ for all $k \in K$ and $\Sigma(\pi_\ell) < \Sigma(\pi)$ for all $\ell \in L$. It also justifies rules of the form

$$\textbf{Form2}\quad \frac{\bigwedge_{k\in K} \pi_k,\ \bigwedge_{\ell\in L}(\pi_\ell\ \text{ implies }\ \pi'_\ell)}{\pi}$$

if $\Sigma(\pi_k) \leq \Sigma(\pi)$ for all $k \in K$, and $\Sigma(\pi_\ell) < \Sigma(\pi)$, $\Sigma(\pi'_\ell) \leq \Sigma(\pi)$ for all $\ell \in L$. Here K and L are (possibly infinite) index sets. However, in both forms of rules, if the index sets are themselves determined by other literals, those literals must be positive and their Σ-values must be strictly less than that of the conclusion π. In other words, if $K = \{k : \pi''_k\}$ then we must have $\Sigma(\pi''_k) < \Sigma(\pi)$ for all $k \in K$. The predicate *inherits* defined further below uses such an index set, and the rules **HideP4** and **HideP5** used in the definition of the hiding construct conform to **Form2** above.

Before proceeding, we give an example as to how the rules should be interpreted. Consider the rules **ECh2** and **EChP1** defined below for the external choice operator $+$:

$$\textbf{ECh2}\ \frac{P[p]{=}a{\Rightarrow}P', \quad Q[q]{\neq}a{\Rightarrow}}{(P+Q)[p+q]{=}a{\Rightarrow}P'} \qquad \textbf{EChP1}\ \frac{nP, \quad \uparrow Q}{n(P+Q)}\ (n \in Nat)$$

The rule **ECh2** should be interpreted as follows:

If it has been inferred that P has an internal state p which offers the action a such that upon performing a, p evolves to P' *and* it has been inferred that Q has an internal state q which does *not* offer the action a (this constitutes a negative premise), *then* it can be inferred that the process $P + Q$ has an an internal state named $p + q$ which offers a such that upon performing a, $p + q$ evolves to P'.

Similarly, the rule **EChP1** should be interpreted as follows:

If it has been inferred that P has the divergence property n *and* it has been inferred that Q is divergent, *then* it can be inferred that the process $P + Q$ has the divergence property n.

Here it is important to note that if an internal state p has been inferred for a process P, then each transition that has been inferred for p is automatically inherited by P. It should therefore be clear that the rule

$$\frac{P[p]}{f(P)[p]}$$

also implies

$$\frac{P[p]{=}a{\Rightarrow}P'}{f(P)[p]{=}a{\Rightarrow}P'}$$

and therefore, the latter does not have to be specified explicitly. Note also that any symbol that appears free in a rule is assumed to be universally quantified outside the *whole* rule.

Now we proceed with the set of rules which specify the operational semantics of MPL.

8.2.2 Universal Rule

This rule makes sure that whenever a weak process enjoys a divergence property n, it also enjoys all divergence properties less than n.

$$\textbf{URule}\ \frac{nP}{mP}\ (m \leq n)$$

8.2.3 Strong Divergence

The constant construct Ω expresses strong divergence. The process Ω has neither an externally observable nor an externally controllable behavior. In fact, we assume that its behavior is completely undefined. Consequently, the construct Ω has neither transition nor internal state rules associated with it. Only the divergence property 0 can be inferred for Ω.

$$\textbf{OmegaP}\ \frac{}{\uparrow\Omega}$$

Here recall that by **URule**, the notations $\uparrow\Omega$ and 0Ω are equivalent.

8.2.4 Termination

Termination is expressed by the constant construct *NIL* which describes a process having a single terminal internal state named s_{NIL}. *NIL* yields a convergent process; consequently, it does not have any rules regarding divergence properties. No transitions can be inferred for s_{NIL}, but unlike Ω, the process *NIL* is assumed to have a well-defined observable behavior: its inability to offer actions to be performed.

$$\textbf{Nil}\ \frac{}{NIL[s_{NIL}]}$$

8.2.5 Prefixing

The family of unary constructs $\{a\cdot : a \in Act\}$ expresses sequentiality. The process $a\cdot P$ is convergent with a single internal state, named $s_{a\cdot P}$, in which the action a is offered. Upon performing a, $a\cdot P$ evolves to P.

$$\textbf{Act}\ \frac{}{(a\cdot P)[s_{a\cdot P}]{=}a{\Rightarrow}P}$$

8.2.6 Hiding

The family of unary constructs $\{\textbf{hide}M : M \subseteq Act\}$ provides a mechanism for abstracting internal behavior. In particular, if $P{=}a{\Rightarrow}Q$ then P **hide** $\{a\}$ has the effect of making the action a invisible to the environment by collapsing the transition on a and merging the set of internal states of P with the set of internal states of Q. Divergence is inferred for P **hide** $\{a\}$ if P can perform an infinite sequence of a-transitions. The divergence properties are *incremented* to take into account weak divergence when P **hide** $\{a\}$ diverges and P has an observable behavior after its immediate a-transitions are hidden (see rule **HideP5**). The same principle is applied recursively to the process Q.

$$\textbf{Hide1}\ \frac{P{=}\bar{a}{\Rightarrow}P'[p']{=}b{\Rightarrow}Q}{(P\,\textbf{hide}\,M)[p'\,\textbf{hide}\,M]{=}b{\Rightarrow}Q\,\textbf{hide}\,M}\ (\bar{a}\in M^*,\ b\notin M)$$

$$\textbf{Hide2}\ \frac{P{=}\bar{a}{\Rightarrow}P'[p'],\quad p'\text{ is terminal}}{(P\,\textbf{hide}\,M)[p']}\ (\bar{a}\in M^*)$$

Before giving the rules regarding the inference of divergence properties, it would be convenient to define the condition under which P **hide** M inherits a divergence property from its M-reachable processes:

Definition 8.1 Let $P\,\mathit{after}\,M \overset{\text{def}}{=} \{P' : P{=}a{\Rightarrow}P' \text{ for some } a\in M\}$. For $n\in \mathit{Nat}$, we say that P *inherits n upon hiding M*, written $\mathit{inherits}(P,n,M)$, if (1) $\uparrow P$ implies nP, and (2) for every $P'\in P\,\mathit{after}\,M$ satisfying $\uparrow(P'\,\textbf{hide}\,M)$, we have $n(P'\,\textbf{hide}\,M)$.

Now we can specify how divergence can be inferred for P **hide** M. This may look more complicated than it really is:

$$\textbf{HideP1}\ \frac{P{=}\rho{\Rightarrow}}{\uparrow(P\,\textbf{hide}\,M)}\ (\rho\colon \mathit{Nat}\longmapsto M)\qquad \textbf{HideP2}\ \frac{\uparrow P}{\uparrow(P\,\textbf{hide}\,M)}$$

$$\textbf{HideP3}\ \frac{P{=}\bar{a}{\Rightarrow}P',\quad \uparrow(P'\,\textbf{hide}\,M)}{\uparrow(P\,\textbf{hide}\,M)}\ (\bar{a}\in M^*)$$

$$\textbf{HideP4}\ \frac{\uparrow(P\,\textbf{hide}\,M),\quad \mathit{inherits}(P,M,n)}{n(P\,\textbf{hide}\,M)}\ (n\in\mathit{Nat},\ n\neq 0)$$

$$\textbf{HideP5}\ \frac{n(P\,\textbf{hide}\,M),\quad \mathit{inherits}(P,M,n),\quad (P\,\textbf{hide}\,M)[s]}{(n+1)(P\,\textbf{hide}\,M)}\ (n\in\mathit{Nat})$$

By the rule **HideP1**, P **hide** M diverges whenever $P{=}\rho{\Rightarrow}$ for some internal computation $\rho\colon \mathit{Nat}\longmapsto M$. The type of divergence (i.e., whether weak or strong) depends on whether termination or progress through an observable action exclusive of M is possible infinitely often along ρ. This is specified by the rules **HideP4** and **HideP5**. P **hide** M converges when no divergence property can be inferred for it.

Note that we can specialize the inference rule **HideP1** to the following:

$$\textbf{HideP1}' \quad \frac{P{=}\bar{a}{\Rightarrow}P}{\uparrow(P\,\textbf{hide}\,M)} \ (\bar{a} \in M^*, \bar{a} \neq \varepsilon)$$

Let us illustrate how the above rules can be applied in the context of two small examples.

Example 1: Inference of strong divergence. Now consider the weak process Q_0 depicted in Figure 16(b). From the figure, we obtain the following transitions and internal states:

$$1.\ Q_0[q_1]{=}b{\Rightarrow}Q_1, \quad 2.\ Q_0[q_2]{=}a{\Rightarrow}Q_2, \quad 3.\ Q_2[q_3]{=}a{\Rightarrow}Q_2, \quad 4.\ Q_1[q_4].$$

Here is how the weak process behavior of $Q_0\,\textbf{hide}\,\{a\}$ can be inferred based upon these previous inferences:

5. $(Q_1\,\textbf{hide}\,\{a\})[q_4\,\textbf{hide}\,\{a\}]$ (by **Hide2**; 4)
6. $(Q_0\,\textbf{hide}\,\{a\})[q_1\,\textbf{hide}\,\{a\}]{=}b{\Rightarrow}Q_1\,\textbf{hide}\,\{a\}$ (by **Hide1**; 1)
7. $0(Q_0\,\textbf{hide}\,\{a\})$ (by **HideP1**; 2, 3)
8. $1(Q_0\,\textbf{hide}\,\{a\})$ (by **HideP5**; 6, 7)

We conclude from line 8 that $\uparrow^s(P_0\,\textbf{hide}\,\{a\})$. The resulting weak process is depicted in Figure 17(b).

Example 2: Inference of weak divergence. Consider the weak process P_0 depicted in Figure 16(a). From the figure, we obtain the following transitions and internal states:

$$1.\ P_0[p_1]{=}a{\Rightarrow}P_0, \quad 2.\ P_0[p_2]{=}b{\Rightarrow}P_1, \quad 3.\ P_1[p_3].$$

Then for $P_0\,\textbf{hide}\,\{a\}$, we can initially infer the following:

4. $(P_1\,\textbf{hide}\,\{a\})[p_3\,\textbf{hide}\,\{a\}]$ (by **Hide2**; 3)
5. $(P_0\,\textbf{hide}\,\{a\})[p_2\,\textbf{hide}\,\{a\}]{=}b{\Rightarrow}P_1\,\textbf{hide}\,\{a\}$ (by **Hide1**; 2)

We infer weak divergence for $P_0\,\textbf{hide}\,\{a\}$ by induction on i-divergence. The basis of the induction is established by the following inference:

6. $0(P_0\,\textbf{hide}\,\{a\})$ (by **HideP1**; 1)

The induction hypothesis is:

7. $n(P_0\,\textbf{hide}\,\{a\})$ (hypothesis)

And finally the induction step is:

8. $(n+1)(P_0\,\textbf{hide}\,\{a\})$ (by **HideP5**; 5, 7)

We conclude that $i(P_0\,\textbf{hide}\,\{a\})$ for every $i \in \mathit{Nat}$; consequently, we have $\uparrow^w(P_0\,\textbf{hide}\,\{a\})$. The resulting weak process is depicted in Figure 17(a).

8.2.7 Internal Choice

The binary construct $\oplus$ models internal nondeterminism. The process $P \oplus Q$ may either behave like P or like Q, but it may not behave like both P and Q simultaneously. The choice here belongs to the system and cannot be influenced by the environment. $P \oplus Q$ converges whenever both P and Q converge. $P \oplus Q$ weakly diverges whenever both P and Q do so, or one of P or Q converges and the other weakly diverges. $P \oplus Q$ strongly diverges if either P or Q strongly diverges.

$$\textbf{ICh1}\ \frac{P[p]}{(P \oplus Q)[p]} \qquad \textbf{ICh2}\ \frac{Q[q]}{(P \oplus Q)[q]}$$

$$\textbf{IChP1}\ \frac{nP, \quad nQ}{n(P \oplus Q)}$$

$$\textbf{IChP2}\ \frac{nP, \quad \downarrow Q}{n(P \oplus Q)} \qquad \textbf{IChP3}\ \frac{\downarrow P, \quad nQ}{n(P \oplus Q)}$$

8.2.8 External Choice

The binary construct $+$ expresses external nondeterminism. The process $P + Q$ can behave both like P and like Q, the choice belonging to the environment. $P + Q$ converges if either P or Q does. $P + Q$ weakly diverges if neither P nor Q converges and one of P or Q weakly diverges. $P + Q$ strongly diverges if both P and Q strongly diverges. The construct $+$ is defined in terms of the internal choice construct $\oplus$.

$$\textbf{ECh1}\ \frac{P[p], \quad Q[q]}{(P + Q)[(p + q)]}$$

$$\textbf{ECh2}\ \frac{P[p]{=}a{\Rightarrow}P', \quad Q[q]{\neq}a{\Rightarrow}}{(P + Q)[(p + q)]{=}a{\Rightarrow}P'} \qquad \textbf{ECh3}\ \frac{P[p]{\neq}a{\Rightarrow}, \quad Q[q]{=}a{\Rightarrow}Q'}{(P + Q)[(p + q)]{=}a{\Rightarrow}Q'}$$

$$\textbf{ECh4}\ \frac{P[p]{=}a{\Rightarrow}P', \quad Q[q]{=}a{\Rightarrow}Q'}{(P + Q)[(p + q)]{=}a{\Rightarrow}P' \oplus Q'}$$

$$\textbf{ECh5}\ \frac{P[p], \quad Int(Q) = \emptyset}{(P + Q)[p]} \qquad \textbf{ECh6}\ \frac{Int(P) = \emptyset, \quad Q[q]}{(P + Q)[q]}$$

$$\textbf{EChP1}\ \frac{nP, \quad \uparrow Q}{n(P + Q)} \qquad \textbf{EChP2}\ \frac{\uparrow P, \quad nQ}{n(P + Q)}$$

8.2.9 Parallel Composition

The family of binary constructs $\{\langle M\rangle : M \subseteq \mathit{Act}\}$ expresses parallel composition based on synchronous communication via the actions inclusive of M and on nondeterministic interleaving of the actions exclusive of M. M is the *the synchronization set* which specifies the actions that must be performed simultaneously by the two processes. Note that the actions belonging to the synchronization set are not automatically hidden; rather parallel composition with automatic hiding is expressed as $(P\langle M\rangle Q)$ **hide** M. Divergence is as in $+$.

$$\textbf{Par1}\ \frac{P[p],\quad Q[q]}{(P\langle M\rangle Q)[(p\langle M\rangle q)]}$$

$$\textbf{Par2}\ \frac{P[p]{=}a{\Rightarrow}P',\quad Q[q]{\neq}a{\Rightarrow}}{(P\langle M\rangle Q)[(p\langle M\rangle q)]{=}a{\Rightarrow}P'\langle M\rangle Q}\ (a \notin M)$$

$$\textbf{Par3}\ \frac{P[p]{\neq}a{\Rightarrow},\quad Q[q]{=}a{\Rightarrow}Q'}{(P\langle M\rangle Q)[(p\langle M\rangle q)]{=}a{\Rightarrow}P\langle M\rangle Q'}\ (a \notin M)$$

$$\textbf{Par4}\ \frac{P[p]{=}a{\Rightarrow}P',\quad Q[q]{=}a{\Rightarrow}Q'}{(P\langle M\rangle Q)[(p\langle M\rangle q)]{=}a{\Rightarrow}(P'\langle M\rangle Q) \oplus (P\langle M\rangle Q')}\ (a \notin M)$$

$$\textbf{Par5}\ \frac{P[p]{=}b{\Rightarrow}P',\quad Q[q]{=}b{\Rightarrow}Q'}{(P\langle M\rangle Q)[(p\langle M\rangle q)]{=}b{\Rightarrow}P'\langle M\rangle Q'}\ (b \in M)$$

$$\textbf{Par6}\ \frac{P[p],\quad \mathit{Int}(Q) = \emptyset}{(P\langle M\rangle Q)[p\langle M\rangle \omega]} \qquad \textbf{Par7}\ \frac{\mathit{Int}(P) = \emptyset,\quad Q[q]}{(P\langle M\rangle Q)[\omega\langle M\rangle q]}$$

$$\textbf{Par8}\ \frac{P[p]{=}a{\Rightarrow}P',\quad \mathit{Int}(Q) = \emptyset}{(P\langle M\rangle Q)[p\langle M\rangle \omega]{=}a{\Rightarrow}(P'\langle M\rangle Q)}\ (a \notin M)$$

$$\textbf{Par9}\ \frac{\mathit{Int}(P) = \emptyset,\quad Q[q]{=}a{\Rightarrow}Q'}{(P\langle M\rangle Q)[\omega\langle M\rangle q]{=}a{\Rightarrow}(P\langle M\rangle Q')}\ (a \notin M)$$

$$\textbf{ParP1}\ \frac{nP,\quad \uparrow Q}{n(P\langle M\rangle Q)} \qquad \textbf{ParP2}\ \frac{\uparrow P,\quad nQ}{n(P\langle M\rangle Q)}$$

8.2.10 Recursive Definitions

Finally, we consider recursive MPL definitions of the form X **def** E. Let us write $E\{X{\leftarrow}E'\}$ to denote the expression obtained by substituting E' for every *free* occurrence of X in E. Similarly, $F\{E{\leftarrow}E'\}$ denotes the expression obtained by substituting E' for every occurrence of E in F. There are three rules governing the construct **def**:

$$\textbf{Def}\ \frac{E\{X{\leftarrow}\Omega_X\}[s]{=}a{\Rightarrow}E'}{(X\ \textbf{def}\ E)[(s\ \textbf{def}\ E)]{=}a{\Rightarrow}E'\{\Omega_X{\leftarrow}(X\ \textbf{def}\ E)\}}$$

$$\mathbf{DefP}\ \frac{nE\{X{\leftarrow}\Omega_X\}}{n(X\ \mathbf{def}\ E)} \qquad \mathbf{DefOmegaP}\frac{}{\uparrow\Omega_X}$$

The rule **Def** is to infer internal states and transitions from a given MPL definition. Here Ω_X is treated the same way as the construct Ω. The subscript X is used to distinguish Ω_X from casual occurrences of Ω in E, so that X **def** E can be substituted back for Ω_X in E'. According to this rule, we can first substitute Ω_X for X in E. Then for every internal state and every transition that may be inferred for the resulting expression, one can infer a corresponding internal state and a corresponding transition for X **def** E. If, as a result, the Ω_X-substituted expression evolves to E', then X **def** E evolves to the expression obtained by substituting itself back for Ω_X in E'.

As far as divergence is concerned, X **def** E inherits it from E. If X is defined in terms of itself, then by the rule **DefP**, Ω_X is substituted for X in E.

Example 3: Unguarded MLP definition with internal choice. Let

$$X\ \mathbf{def}\ X \oplus a{\cdot}X$$

Here is how we can infer the weak process behavior of this recursive MPL definition in which X is bound but has an occurrence which is not always guarded by a prefixing construct.

1. $(a{\cdot}\Omega)[s_{a\cdot\Omega}]{=}a{\Rightarrow}\Omega$ (by **Act**)
2. $(\Omega \oplus a{\cdot}\Omega)[s_{a\cdot\Omega}]$ (by **ICh1**; 1)
3. $(X\ \mathbf{def}\ X \oplus a{\cdot}X)[(s_{a\cdot\Omega}\ \mathbf{def}\ X \oplus a{\cdot}X)]{=}a{\Rightarrow}X\ \mathbf{def}\ X \oplus a{\cdot}X$ (by **Def**; 2)
4. $0(\Omega \oplus a{\cdot}\Omega)$ (by **OmegaP**; **IChP2**)
5. $0(X\ \mathbf{def}\ X \oplus a{\cdot}X)$ (by **DefP**; 4)

Example 4: External choice, parallel composition, and hiding. Now consider

$$((X\ \mathbf{def}\ a{\cdot}X + t{\cdot}b{\cdot}X)\langle t\rangle(Y\ \mathbf{def}\ t{\cdot}Y))\ \mathbf{hide}\ \{t\}$$

For convenience, let us abbreviate some of the sub-expressions of this MPA expression as follows:

$$\underbrace{(\underbrace{(X\ \mathbf{def}\ \underbrace{a{\cdot}X + t{\cdot}b{\cdot}X}_{W})}_{P}\langle t\rangle\underbrace{(Y\ \mathbf{def}\ \underbrace{t{\cdot}Y}_{Z})}_{Q})\ \mathbf{hide}\ \{t\}}_{R}$$

The overall expression is abbreviated by R. The MPL expressions P, Q, and R correspond to the three processes (having the same names) which were discussed informally in the time-out example of Section 2.4.

1. $(a{\cdot}\Omega)[s_{a\cdot\Omega}]{=}a{\Rightarrow}\Omega$ (by **Act**)
2. $(b{\cdot}\Omega)[s_{a\cdot\Omega}]{=}b{\Rightarrow}\Omega$ (by **Act**)
3. $(t{\cdot}\Omega)[s_{a\cdot\Omega}]{=}t{\Rightarrow}\Omega$ (by **Act**)
4. $(t{\cdot}b{\cdot}\Omega)[s_{t\cdot b\cdot\Omega}]{=}t{\Rightarrow}b{\cdot}\Omega$ (by **Act**)
5. $Q[(s_{t\cdot\Omega}\ \mathbf{def}\ Z)]{=}t{\Rightarrow}Q$ (by **Def**; 3)
6. $(a{\cdot}\Omega + t{\cdot}b{\cdot}\Omega)[(s_{a\cdot\Omega} + s_{t\cdot b\cdot\Omega})]{=}a{\Rightarrow}\Omega$ (by **ECh2**; 1, 4)
7. $(a{\cdot}\Omega + t{\cdot}b{\cdot}\Omega)[(s_{a\cdot\Omega} + s_{t\cdot b\cdot\Omega})]{=}t{\Rightarrow}b{\cdot}\Omega$ (by **ECh3**; 1, 4)
8. $P[((s_{a\cdot\Omega} + s_{t\cdot b\cdot\Omega})\ \mathbf{def}\ W)]{=}t{\Rightarrow}b.P$ (by **Def**; 7)
9. $P[((s_{a\cdot\Omega} + s_{t\cdot b\cdot\Omega})\ \mathbf{def}\ W)]{=}a{\Rightarrow}P$ (by **Def**; 6)
10. $(b{\cdot}P)[s_{b\cdot P}]{=}b{\Rightarrow}P$ (by **Act**)
11. $(P\langle t\rangle Q)[(((s_{a\cdot\Omega} + s_{t\cdot b\cdot\Omega})\ \mathbf{def}\ W)\langle t\rangle(s_{t\cdot\Omega}\ \mathbf{def}\ Z))]{=}a{\Rightarrow}P\langle t\rangle Q$ (by **Par2**; 5, 9)
12. $(P\langle t\rangle Q)[(((s_{a\cdot\Omega} + s_{t\cdot b\cdot\Omega})\ \mathbf{def}\ W)\langle t\rangle(s_{t\cdot\Omega}\ \mathbf{def}\ Z))]{=}t{\Rightarrow}b{\cdot}P\langle t\rangle Q$ (by **Par5**; 5, 8)
13. $(b{\cdot}P\langle t\rangle Q)[(s_{b\cdot P}\langle t\rangle(s_{t\cdot\Omega}\ \mathbf{def}\ Z))]{=}b{\Rightarrow}P\langle t\rangle Q$ (by **Par2**; 5, 10)
14. $R[(((s_{a\cdot\Omega} + s_{t\cdot b\cdot\Omega})\ \mathbf{def}\ W)\langle t\rangle(s_{t\cdot\Omega}\ \mathbf{def}\ Z))\ \mathbf{hide}\ \{t\}]{=}a{\Rightarrow}R$ (by **Hide1**; 11)
15. $R[(s_{b\cdot P}\langle t\rangle(s_{t\cdot\Omega}\ \mathbf{def}\ Z))\ \mathbf{hide}\ \{t\}]{=}b{\Rightarrow}R$ (by **Hide1**; 13)

It is easy to see that the resulting weak process is indeed strongly equivalent to the weak process defined by the much simpler MPL expression $R\ \mathbf{def}\ a{\cdot}R \oplus b{\cdot}R$.

9 Conclusions and Discussion

There appears to be support in the literature for the view that internal transitions should not be a part of a language and that they should not be explicit in a semantic theory of nondeterminism and concurrency. The several behavioral equivalences which attempt to compensate for the presence of internal transitions, the notion of derived transition system [7, 29, 21], the numerous denotational theories which do not

$$\frac{}{(a;P)[\diamond p_{a;P}]{=}a{\Rightarrow} P} \qquad \frac{P[\diamond p]}{(\mathbf{i};P)[\bullet p]} \qquad \frac{P[\bullet p]}{(\mathbf{i};P)[\bullet p]}$$

$$\frac{P[\bullet p]{=}a{\Rightarrow} P', \quad Q[\diamond q]{\neq}a{\Rightarrow}}{(P \mathbin{[]} Q)[\diamond(\bullet p \mathbin{[]} \diamond q)]{=}a{\Rightarrow} P'} \qquad \frac{P[\bullet p]{\neq}a{\Rightarrow}, \quad Q[\diamond q]{=}a{\Rightarrow} Q'}{(P \mathbin{[]} Q)[\diamond(\bullet p \mathbin{[]} \diamond q)]{=}a{\Rightarrow} Q'}$$

$$\frac{P[\diamond p]{=}a{\Rightarrow} P', \quad Q[\bullet q]{\neq}a{\Rightarrow}}{(P \mathbin{[]} Q)[\diamond(\diamond p \mathbin{[]} \bullet q)]{=}a{\Rightarrow} P'} \qquad \frac{P[\diamond p]{\neq}a{\Rightarrow}, \quad Q[\bullet q]{=}a{\Rightarrow} Q'}{(P \mathbin{[]} Q)[\diamond(\diamond p \mathbin{[]} \bullet q)]{=}a{\Rightarrow} Q'}$$

$$\frac{P[\diamond p]{=}a{\Rightarrow} P', \quad Q[\diamond q]{\neq}a{\Rightarrow}}{(P \mathbin{[]} Q)[\diamond(\diamond p \mathbin{[]} \diamond q)]{=}a{\Rightarrow} P'} \qquad \frac{P[\diamond p]{\neq}a{\Rightarrow}, \quad Q[\diamond q]{=}a{\Rightarrow} Q'}{(P \mathbin{[]} Q)[\diamond(\diamond p \mathbin{[]} \diamond q)]{=}a{\Rightarrow} Q'}$$

$$\frac{P[\bullet p]{=}a{\Rightarrow} P', \quad Q[\diamond q]{=}a{\Rightarrow} Q'}{(P \mathbin{[]} Q)[\diamond(\bullet p \mathbin{[]} \diamond q)]{=}a{\Rightarrow}\mathbf{i};P' \mathbin{[]} \mathbf{i};Q'} \qquad \frac{P[\diamond p]{=}a{\Rightarrow} P', \quad Q[\bullet q]{=}a{\Rightarrow} Q'}{(P \mathbin{[]} Q)[\diamond(\diamond p \mathbin{[]} \bullet q)]{=}a{\Rightarrow}\mathbf{i};P' \mathbin{[]} \mathbf{i};Q'}$$

$$\frac{P[\diamond p]{=}a{\Rightarrow} P', \quad Q[\diamond q]{=}a{\Rightarrow} Q'}{(P \mathbin{[]} Q)[\diamond(\diamond p \mathbin{[]} \diamond q)]{=}a{\Rightarrow}\mathbf{i};P' \mathbin{[]} \mathbf{i};Q'} \qquad \frac{P[\bullet p]}{(P \mathbin{[]} Q)[\bullet p]} \qquad \frac{Q[\bullet q]}{(P \mathbin{[]} Q)[\bullet q]}$$

Figure 18: Weak process semantics of LOTOS's choice ([]), internal action (**i**;), and prefix (a;) constructs.

refer to internal transitions at all, the motivation behind the τ-less version of CCS discussed in [9], can be considered as indications of such support.

In this paper, we introduced Abstract Transition Systems — an effective, internal transition-free operational model for nondeterministic concurrent processes. The main concept developed was that of a weak process in which the familiar, powerful notion of internal transition is replaced by a less powerful, yet effective notion: the internal state. We have also discussed divergence, the ability of a system to undertake an infinite internal computation, from an operational perspective. With a simple extension to the basic the ATS model, we were able to distinguish between pathological (strong) and non-pathological (weak) forms of divergence — a distinction which has not been addressed elsewhere. The theory of divergence presented is novel. It gives rise to an an elegant "limit" interpretation which allows weak divergence to be inferred from strong divergence using mathematical induction. The applicability of the extended ATS (EATS) model has been demonstrated by assigning operational semantics to a small process algebra.

The concept of weak process brings internal nondeterminism to the foreground. This makes the EATS model suitable for process algebraic languages which express internal nondeterminism by means of an internal choice construct (e.g., TCSP [5], the τ-less version of CCS proposed in [9], and the process algebra described in [14]). Nonetheless, the EATS model can also be used in assigning operational semantics to such languages as CCS [20], LOTOS [27], and ACP [3] which supply an internal action construct. With a weak process (or EATS-based) semantics, the complexity underlying internal nondeterminism is dealt with by the operational definitions of the relevant constructs themselves. By contrast, with an LTS-based semantics, this complexity is usually deferred to a higher level, where it is resolved by a behavioral equivalence.

As an example consider LOTOS, where the strong process semantics of the hiding and the internal action constructs are almost trivial:

$$\frac{P{-}a{\rightarrow}P'}{\textbf{hide}\ M\ \textbf{in}\ P\ {-}\tau{\rightarrow}\ \textbf{hide}\ M\ \textbf{in}\ P'}\ (a \in M)$$

$$\frac{P{-}b{\rightarrow}P'}{\textbf{hide}\ M\ \textbf{in}\ P\ {-}b{\rightarrow}\ \textbf{hide}\ M\ \textbf{in}\ P'}\ (b \notin M) \qquad \frac{}{\mathbf{i};P{-}\tau{\rightarrow}P}$$

This simplicity, however, is misleading. Here the complexity underlying these constructs is not actually dealt with by the above semantic rules: rather, it is the responsibility of the adopted LTS behavioral equivalence, such as testing equivalence [4], to abstract from internal behavior, and to handle the complexity resulting from this type of abstraction. Without such an equivalence, no useful semantic theory can be obtained. The behavioral equivalence specifies the actual semantics underlying the internal action construct by taking into account the presence of internal transitions, and therefore, it indirectly defines the semantics underlying hiding. The same argument is also valid for CCS's τ and restriction constructs *vis-à-vis* observation (or weak bisimulation) equivalence. It seems like in an operational theory, simple semantic rules are possible at the expense of a complex behavioral equivalence. This equivalence usually relies on the notion of *derived transition relation* [29] (an abstraction of the original transition relation with respect to internal transitions), and therefore, it cannot be computed by using only "local" information. An EATS-based semantics shifts this complexity to the operational definitions of the relevant constructs at the advantage of a simpler behavioral equivalence which can be formulated inductively using only "local" information. A major disadvantage here is the use of negative premises in the rules. If negative premises are employed, they must be justified, for example using the stratification method proposed by Groote [13].

It is not our intention to argue that explicit internal transitions should be avoided at all cost. However the model of weak processes demonstrates that they can be avoided at the expense of simple semantic rules. This disadvantage is compensated for by two advantages: (1) the ability to adopt a simple, straightforward behavioral equivalence and (2) the ability to separate the semantics underlying such inherently complex constructs as hiding from the adopted behavioral equivalence. The latter is the main reason weak process-based operational semantics result in relatively complicated inference rules for such constructs as hiding.

A quick look at the choice and internal action constructs of LOTOS reveals just how complicated the inference rules can get. In LOTOS, the choice ([]) and the internal action (**i**;) constructs interact, and thus they cannot be defined independently in the EATS model. The same can also be said for the analogous CCS constructs $+$ and τ. This interdependency can be seen in Figure 18. The weak process semantics proposed in the figure is consistent with testing equivalence [4, 7] in that it is possible to envision a behavioral equivalence on weak processes which equates two LOTOS behaviors if and only if they are testing equivalent. Unlike coarse equivalence, testing equivalence derives from a notion of external testing based on interpretation (I), rather than on interpretation (II), of internal nondeterminism discussed in Section 4. Note how a naming scheme using the internal state labels $\bullet$ and $\diamond$ is employed in the inference rules to express the dependency between the choice and the internal action constructs.

The results of Section 7.4 imply that even after having added divergence properties to the basic ATS model, the concept of weak process remains less powerful than the concept of strong process. However, what is more interesting is that the expressiveness results seem to hold true even when the strongest perspective of external behavior for the strong process model is relaxed from $\sim$ to a divergence discriminating version of weak bisimulation (observation) equivalence $\approx$.[4] The refined version of $\approx$ we consider — which we may denote by $\approx^{\uparrow}$ — is defined in a similar way to $\sim_c^{\uparrow}$, and is stronger than the refinements discussed in [29]. We postulate that such an equivalence, $\approx^{\uparrow}$, would be stronger than $\sim_c^{\uparrow}$. Consequently, as for $\sim$, it would be impossible to capture $\approx^{\uparrow}$ within the framework of the EATS model. The implications of this conjecture are twofold. First, no weak process semantics of CCS can be fully consistent with (or isomorphic to) its standard strong process semantics defined in terms of observation equivalence. Second, it suggests that observation equivalence does not totally abstract from internal transitions, and therefore, is too strong.

[4]For weak bisimulation equivalence, refer to [21] and [29].

Acknowledgements—Many thanks to Prof. Rance Cleaveland for his detailed comments and invaluable criticisms on an early version of this document which appeared as a chapter in the first author's PhD thesis.

References

[1] Abramski, S. Observation equivalence as a testing equivalence. *Theoretical Computer Science 53* (1987), 225–241.

[2] Baeten, J. C. M., and Klop, J. W., Eds. *CONCUR '90 — Theories of Concurrency: Unification and Extension* (1990), no. 458 in Lecture Notes in Computer Science, Springer-Verlag.

[3] Bergstra, J. A., and Klop, J. W. Algebra of communicating processes with abstraction. *Theoretical Computer Science 37* (1985), 77–121.

[4] Brinksma, E. On the existence of canonical testers. Memorandum INF-87-5, Department of Informatics, University of Twente, Netherlands, 1987.

[5] Brookes, S. D., Hoare, C. A. R., and Roscoe, A. W. A theory of Communicating Sequential Processes. *Journal of the ACM 31*, 3 (July 1984), 560–599.

[6] Brookes, S. D., and Roscoe, A. W. An improved failures model for communicating processes. In *Seminar on Concurrency* (1984), S. D. Brookes, A. W. Roscoe, and G. Winskel, Eds., no. 197 in Lecture Notes in Computer Science, Springer-Verlag.

[7] De Nicola, R. Extensional equivalences for transition systems. *Acta Informatica 24* (1987), 211–237.

[8] De Nicola, R., and Hennessy, M. Testing equivalences for processes. *Theoretical Computer Science 34* (1984), 83–133.

[9] De Nicola, R., and Hennessy, M. CCS without τs. In *Proceedings of Tapsoft'87,International Conference on Theory and Practice of Software Development* (1987), no. 250 in Lecture Notes in Computer Science, Springer-Verlag.

[10] Erdogmus, H. *A Flexible Framework for the Design of Concurrent Nondeterministic Processes.* PhD thesis, INRS-Télécommunications, Verdun, Québec, 1993.

[11] Erdogmus, H., and Johnston, R. On the specification and synthesis of communicating processes. *IEEE Transactions on Software Engineering 16*, 12 (Dec. 1990), 1412–1426.

[12] Fournier, R., and von Bochmann, G. The equivalence in the DCP model. *Theoretical Computer Science 87* (1991), 97–114.

[13] Groote, J. F. Transition system specifications with negative premises. In Baeten and Klop [2], pp. 332–341.

[14] Hennessy, M. *Algebraic Theory of Processes.* MIT Press, Cambridge, MA, 1988.

[15] Hennessy, M., and Milner, R. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM 32*, 1 (Jan. 1985).

[16] Hoare, C. A. R. Communicating Sequential Processes. *Communications of the ACM 21*, 8 (Aug. 1978), 666–667.

[17] Keller, R. Formal verification of parallel programs. *Communications of the ACM 19* (1976), 561–572.

[18] Larsen, K. G. A context dependent bisimulation between processes. *Theoretical Computer Science 49* (1987), 185–215.

[19] Leduc, G. *On the Role of Implementation Relations in the Design of Distributed Systems using LOTOS.* Thèse d'agréation de l'enseignement supérieur, Faculté des sciences appliquées, Université de Liège, Belgium, June 1991.

[20] Milner, R. *A Calculus of Communicating Systems.* No. 92 in Lecture Notes in Computer Science. Springer-Verlag, 1980.

[21] Milner, R. *Communication and Concurrency.* Prentice-Hall, 1989.

[22] Nielson, H. R., and Nielson, F. *Semantics with Applications — A Formal Introduction.* Lecture Notes in Computer Science. John Wiley & Sons, England, 1992, ch. 2.

[23] Olderog, E. R., and Hoare, C. A. R. Specification-oriented semantics for communicating processes. *Acta Informatica 23* (1986), 9–66.

[24] Park, D. M. R. Concurrency and automata for infinite sequences. In *Proceedings of 5th GI Conference* (1981), no. 104 in Lecture Notes in Computer Science, Springer-Verlag.

[25] Plotkin, G. A structural approach to operational semantics. Daimi-fn-19, Computer Science Department, Aarhus University, Denmark, 1981.

[26] Rotman, B., and Kneebone, G. *The Theory of Sets and Transfinite Numbers.* Oldbourne, London, 1966.

[27] van Eijk, P. H. J., Vissers, C. A., and Diaz, M., Eds. *The Formal Description Technique LOTOS*. North Holland, 1989.

[28] van Glabbeek, R. J. The linear time – branching time spectrum. In Baeten and Klop [2], pp. 278–297.

[29] Walker, D. Bisimulation and divergence. In *Proceedings of IEEE Symposium on Logic in Computer Science* (1988), pp. 186–191.

[30] Zuckerman, M. *Sets and Transfinite Numbers*. MacMillan, New York, 1974.