Theoretical Computer Science 75 (1990) 223-262
North-Holland

FUNDAMENTAL STUDY

A PARTIAL ORDERING SEMANTICS FOR CCS

P. DEGANO
Dipartimento di Informatica, Università di Pisa, Italy

R. DE NICOLA
Istituto di Elaborazione dell'Informazione, C.N.R., Pisa, Italy

U. MONTANARI
Dipartimento di Informatica, Università di Pisa, Italy

Communicated by J. de Bakker
Received May 1988
Revised November 1988

Abstract. A new operational semantics for "pure" CCS is proposed that considers the parallel operator as a first class one, and permits a description of the calculus in terms of partial orderings. The new semantics (also for unguarded agents) is given in the SOS style via the *partial ordering derivation relation.* CCS agents are decomposed into sets of sequential subagents. The new derivations relate sets of subagents, and describe their actions and the causal dependencies among them. The computations obtained by composing partial ordering derivations are "observed" either as interleaving or partial orderings of events. Interleavings coincide with Milner's many step derivations, and "linearizations" of partial orderings are *all and only* interleavings. Abstract semantics are obtained by introducing two relations of observational equivalence and congruence that preserve concurrency. These relations are finer than Milner's in that they distinguish interleaving of sequential nondeterministic agents from their concurrent execution.

Contents

1. Introduction

Many different models have been proposed to describe systems whose subparts can progress in parallel, synchronize and exchange messages. These models can be

0304-3975/90/$03.50

compared by examining the way in which they describe the fact that events (atomic actions, synchronizations, communications) can be performed concurrently by subparts of a system, i.e., independently from one to another. If we take this standpoint, the various models of concurrency can be divided into two broad groups: those based on *interleaving* and those based on *true concurrency.*

Models based on interleaving express concurrency among events by saying that they may occur in any order. Thus, a total ordering among possibly spatially separated and causally independent events is imposed: a *global clock* and *global states* are assumed. The proposers of such models, which include [27, 28, 32, 25, 1, 3, 4, 15, 30, 22], stress the simplicity of the underlying mathematics as a sufficient reason to advocate this approach, since it permits easier reasoning about concurrent systems and proving most of their properties.

On the other hand, models based on true concurrency use partial ordering of events where concurrency is represented as absence of ordering. Within this framework, *no global clock* is assumed and the behaviour of a system is expressed in terms of the causal relations between the events performed by subparts of its *distributed state.* Their proposers (see for example [26, 24, 36, 40, 31, 39, 16, 20, 5, 38, 2, 17, 18]) claim that these models offer a more faithful picture of reality, and that certain liveness properties of concurrent systems can be better understood and studied within this framework.

A classical representative of models based on interleaving is Milner's Calculus of Communicating Systems (CCS) [27]. It relies on a small number of operators which are used to build terms. These are considered as agents which, by performing certain actions, will become other agents. The operational semantics of the calculus is given through labelled transition systems, and the fact that agent E_0 evolves to E_1 by performing an action μ is rendered by $E_0 \rightarrow^{\mu} E_1$. The technique used (Structured Operational Semantics or SOS [37]) relies on the well-known idea of describing the behaviour of systems by sequences of transitions between configurations. Transitions of compound systems are defined in a syntax-driven way, via axioms and inference rules.

Since the original version of CCS was geared towards the interleaving approach, its semantics does not consider the operator for parallel composition of processes "|" as primitive: given any finite process containing |, there will always exist another process without | which exhibits the same behaviour.

This paper proposes a new operational semantics for CCS that considers the parallel operator as a first class operator, and offers a partial ordering semantics for the calculus. The operational semantics is still given in the SOS style, but a different transition relation, called the *partial ordering derivation relation*, is defined. This relates subparts of CCS agents, rather than their whole global state, and carries information about causal dependencies. CCS agents are decomposed into sets of sequential processes, called *grapes*, and the new transitions not only describe the actions agents may perform when in a given state, but they also express the causal relation among subparts of agents when the global state changes. The partial ordering derivation relation is defined via inference rules which directly correspond to those of [27]. Consequently, the deduction of both transitions follows the same pattern.

The new transitions have the form $I_1 \rightarrow^{[\mu, \mathscr{R}]} I_2$ where I_1 and I_2 represent sets of grapes, and $\mathscr{R}$ is a relation providing additional information about the causal relations among agents. The grapes in I_1 perform the action μ and evolve to those in I_2. We thus say that the grapes of I_1 *cause* those in I_2, through μ. Information about other grapes caused by grapes in I_1, but not through μ, is recorded in $\mathscr{R}$. The intended dynamic meaning is that, after showing an event labelled by μ, the set of grapes I_1, occurring in the current state, can be replaced by the grapes in I_2 and by those related to I_1 via $\mathscr{R}$, thus obtaining the new state.

As an example, consider the CCS agent $(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL}$, which may evolve to $\mathrm{NIL}|\beta.\mathrm{NIL}$ after resolving the nondeterministic choice (expressed by +) in favour of α. In the interleaving approach, this will be rendered as

$$(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL} \xrightarrow{\alpha} \mathrm{NIL}|\beta.\mathrm{NIL}. \qquad (*)$$

We will write it as

$$\{(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL}\} \xrightarrow{[\alpha,\{(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL} \leq \mathrm{id}|\beta.\mathrm{NIL}\}]} \{\mathrm{NIL}|\mathrm{id}\} \qquad (**)$$

where $(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL}$, $\mathrm{NIL}|\mathrm{id}$ and $\mathrm{id}|\beta.\mathrm{NIL}$ are grapes.

In this way, we describe the fact that grape $(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL}$ causes both grape $\mathrm{id}|\beta.\mathrm{NIL}$ and the event labelled by α which in turn causes grape $\mathrm{NIL}|\mathrm{id}$. Note that the possibility that $\mathrm{id}|\beta.\mathrm{NIL}$ may have to perform β independently of the occurrence of α is implied by the absence of any causal relation between α and $\mathrm{id}|\beta.\mathrm{NIL}$. The α-derivation of grape $(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL}$ is shown in Fig. 1. It should be noted that every derivation of the original calculus can always be recovered from our partial ordering derivation simply by "putting together" its initial and final sets of graphs. In the example above, we obtain $\mathrm{NIL}|\beta.\mathrm{NIL}$ by putting together the two grapes $\mathrm{NIL}|\mathrm{id}$ and $\mathrm{id}|\beta.\mathrm{NIL}$.

A transition of the above form may look a bit unnatural. We are used to conceiving labelled transitions as relations between a set of processes and an action, and between that action and *all* the new processes. Instead, in the transition (**) above,

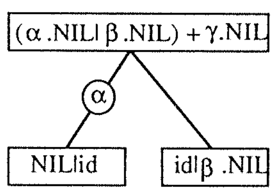
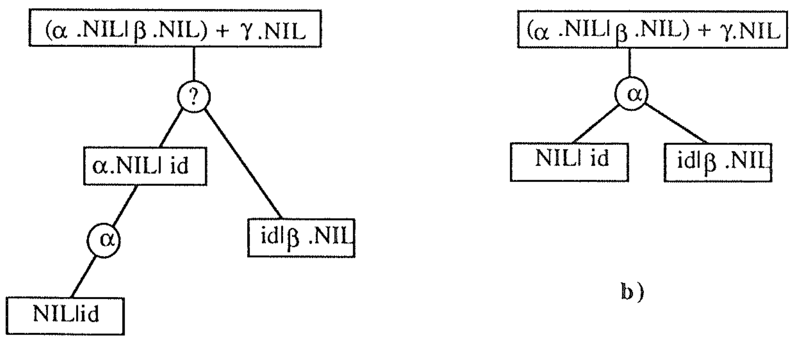


Fig. 1. The transition of the partial ordering operational semantics $\{(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL})\} \rightarrow^{[\alpha,\{(\alpha.\mathrm{NIL}|\beta.\mathrm{NIL})+\gamma.\mathrm{NIL} \leq \mathrm{id}|\beta.\mathrm{NIL}\}]} \{\mathrm{NIL}|\mathrm{id}\}$. Grapes are represented by labelled boxes, events by labelled circles and the causal relation is expressed through its Hasse diagram growing downwards.

grape id $|$ β.NIL is directly related to grape $(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}$. This happens because the evolution of this type of nondeterministic processes requires that first one of the alternatives is chosen, and an action of the chosen grapes is performed. A possible way of describing the above α-transition is illustrated in Fig. 2a. First, a choice-event causes two concurrent grapes $\alpha.\text{NIL}|\text{id}$ and $\text{id}|\beta.\text{NIL}$; the former then performs an α. It is however important to note that, in order to be faithful to the original semantics, the decision and the action can only be considered as a single indivisible action. Since CCS has no mechanisms for defining atomic actions from sequences, we are left with two alternatives. The first requires hiding inside the source grape the decision to obtain transitions such as those of Fig. 1. We would like to stress that this discussion is just for the sake of clarity and does not imply at all introducing any invisible action whatsoever in our semantics. The second alternative is to incorporate the decision into the action itself to obtain the usual transitions (Fig. 2b). In [8, 10], we have followed the latter approach, but it results in an operational semantics that does not take the possible parallelism of CCS agents fully into account. For example, independencies are lost between some concurrent actions in +-context; in the case of the agent $(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}$, a causal relation between α and β is enforced, thus identifying this agent with $\alpha.\beta.\text{NIL}+\beta.\alpha.\text{NIL}+\gamma.\text{NIL}$. A third approach, followed in [11] and [34] introduces a new decomposition according to which the agent $(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}$ originates two grapes, namely $(\alpha.\text{NIL}|\text{id})+\gamma.\text{NIL}$ and $(\text{id}|\beta.\text{NIL})+\gamma.\text{NIL}$. These papers will be further discussed later in this section and in the concluding one.



a)

Fig. 2. Alternative descriptions of the α-transition of agent $(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}$.

A *computation* is a sequence of sets of grapes (i.e., system states corresponding to CCS agents), and of partial ordering derivations (i.e., system transitions). A computation of agent $(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}$ is

$$\xi=\{(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}\}$$

$$\{(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}\}\xrightarrow{[\alpha,\{(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}\leqslant\text{id}|\beta.\text{NIL}\}]}\{\text{NIL}|\text{id}\}$$

$$\{\text{NIL}|\text{id}, \text{id}|\beta.\text{NIL}\}$$

$$\{\text{id}|\beta.\text{NIL}\} \xrightarrow{[\beta,\emptyset]} \{\text{id}|\text{NIL}\}$$

$$\{\text{NIL}|\text{id}, \text{id}|\text{NIL}\}.$$

From computations, we can extract either sequences or partial orderings of actions. In the first case, we keep track of the temporal ordering in which the actions have been performed (in our example α *followed* by β, i.e., $\alpha\beta$); while in the second case we keep track of the causal dependencies among the actions (in our example α *concurrent* with β). By "observing" computations in either way and by taking into account their initial and final sets of grapes, we obtain *interleaving* or *partial ordering many step derivations*. Incidentally, we observe that our approach is indeed operational since we build our many step derivations by composing *elementary* steps and then abstracting. This differs from other approaches [e.g., 2, 5], in which transition systems are used to directly associate partial orderings to agents: the notion of elementary step and the possibility of growing computations from them are lost in favour of a more denotational style.

These two kinds of derivations provide us with a firm ground for studying the relationships between the interleaving and the partial ordering approaches. The natural direct correspondence between our partial ordering derivation relation and Milner's allows us to prove that his many step derivations coincide with our interleaving derivations. This result also guarantees that the original interleaving operational semantics of CCS is immediately retrievable from the partial ordering one. Furthermore, we will show that "linearizing" the causal relation of the partial ordering many step derivation of a computation results in the set of sequences which are *all and only* the interleaving many step derivations. In other words, given a partial ordering of events (obtained from a computation) and a total ordering $\leqslant$ compatible with it, it is always possible to find a computation the events of which are generated exactly as demanded by $\leqslant$. This property, which is called *complete concurrency* in [10], plays a crucial role in relating the interleaving and partial ordering semantics of CCS, namely in proving Theorems 4.9, 4.13, 4.14, 4.17. Returning to our example, all and only the linearizations of the partial ordering of the derivation obtained from ξ (α concurrent with β) are exactly Milner's many step derivations associated with agent $(\alpha.\text{NIL}|\beta.\text{NIL})+\gamma.\text{NIL}$ (when the same side of the + is chosen), i.e., $\alpha\beta$ and $\beta\alpha$.

When the behaviour of concurrent systems is described through a relation between their states, all their internal states must be taken into account. Frequently, however, only some of these states are actually relevant for system analysis. Thus, operational descriptions of this kind end up specifying too many details, and introducing unnecessary and unnatural differentiations. A remedy advocated by Milner is to consider concurrent systems as black boxes, to assume certain actions as internal, thus invisible, and therefore to describe system behaviour only in terms of vibisible actions. For this purpose, notions of observational equivalence and congruence based on experimentations are introduced which make it possible to abstract from unwanted details [27, 29, 23]. Because of the intrinsically sequential nature of the experiments allowed, concurrency is still not a primitive notion of the theory. Here, we introduce a new notion of partial ordering observation which we can use to define notions of observational equivalence and congruence that preserve concurrency.

Like Milner, our starting point is the notion of bisimulation [35]: two agents are equivalent if they are able to perform the same partial orderings of visible actions, evolving to equivalent agents. The new relations of partial ordering observational equivalence and congruence are finer than Milner's in that they distinguish interleaving of sequential nondeterministic processes from their concurrent execution. The two equivalences and the two congruences coincide when dealing only with nondeterministic sequential processes.

We began our investigation on a partial ordering approach to the semantics of concurrent languages some years ago, and our intermediate results have been reported in a number of papers [8, 9, 10, 14, 17, 18]. However, as already mentioned, the semantics for CCS proposed in the first three papers is not completely satisfactory. In fact, we had kept a one-to-one correspondence between the set of grapes reachable through derivations and agents, between the new rules and Milner's, and between the proofs of the derivations, but we did not always permit the concurrent execution of intuitively independent actions. In [11] we solve this problem at the price of a more complex notion of distributed state, and of a less natural set of rules. Actually, due to a distributed treatment of the choice operator, a decomposition relation is introduced which causes the loss of the one-to-one correspondence between states, i.e., sets of grapes, and CCS agents. More detailed comments can be found in Section 3, after Definition 3.2.

In this paper, we are able to give a full account of parallelism while maintaining a syntactic one-to-one correspondence between the interleaving and partial ordering approaches. We keep a centralized treatment of choice thus avoiding state explosion. Hence, the solution proposed here is more suited when there is no real need for distributing choices, and whenever there are space or time constraints. As a matter of fact, a completely distributed implementation, as the one suggested in [11, 34], will require introducing rather sophisticated protocols. Moreover, the present approach straightforwardly deals with unguarded recursion. The causal relation among events may in this case be infinitely branching, thus reflecting the possible unbounded parallelism (see also Fig. 8).

The rest of the paper is organized as follows. Section 2 surveys the original interleaving semantics of CCS which relies on the derivation relation and on the notion of bisimulation. Section 3 defines the new partial ordering derivation relation on sets of subagents rather than on whole agents. The partial ordering many step derivation relation is introduced in Section 4 and compared with Milner's. Using this new relation, partial ordering observational equivalence and congruence are defined in the same section, and shown to be finer than the originals, yet concident on sequential nondeterministic agents. Finally, Section 5 discusses the relationship between this work and other proposals of truly concurrent semantics for CCS.

2. CCS and its interleaving semantics

This section contains a brief introduction to "pure" CCS, i.e., the calculus without value passing. First, we shall introduce the syntax of the calculus, then we will present the traditional interleaving semantics and the observational equivalence and congruence of [27] as refined in [29].

Definition 2.1 (*agents*). Let

- $\Delta = \{\alpha, \beta, \gamma, \dots\}$ be a fixed set and $\Delta^- = \{\alpha^- \mid \alpha \in \Delta\}$, assuming $(\alpha^-)^- = \alpha$;
- $\Lambda = \Delta \cup \Delta^-$ (ranged over by λ) be the set of *visible actions*;
- $\tau \notin \Lambda$ be a distinguished *invisible action*, and let $\Lambda \cup \{\tau\}$ be ranged over by μ.

The CCS agents, ranged over by E, consists of all closed terms (i.e., terms without free variables) which can be generated by the following BNF-like grammar

$$E ::= x \mid \mathrm{NIL} \mid \mu.E \mid E\backslash\alpha \mid E[\phi] \mid E+E \mid E \mid E \mid \mathrm{rec}\, x.\, E,$$

where x is a variable and ϕ is a permutation of $\Lambda \cup \{\tau\}$ which preserves τ and the operation $^-$ of complementation. We assume that the precedence among operators is $\backslash\alpha > [\phi] > \mu. > \mathrm{rec} > + > \mid$.

CCS has a two level semantics: the first level describes the behaviour of agents through an abstract machine and the second level forgets their internal structure by identifying those machines which all exhibit the same external behaviour.

The first level, i.e., the interleaving operational semantics, is based on a labelled transition system with a transition relation defined via a set of transition rules. The relation, called *derivation relation* and denoted by $\to^{\mu}$, relies on the intuition that agent E_0 may evolve to become agent E_1 either by reacting to a λ-stimulus from its environment ($E_0 \to^{\lambda} E_1$) or by performing an internal action which is independent of the environment ($E_0 \to^{\tau} E_1$).

Definition 2.2 (*transitions*). *Milner's derivation relation* $E_0 \to^{\mu} E_1$ is defined as the least relation satisfying the following axiom and inference rules.

(Act) $\mu E \to^{\mu} E$.
(Res) $E_0 \to^{\mu} E_1$ implies $E_0\backslash\alpha \to^{\mu} E_1\backslash\alpha$, $\mu \notin \{\alpha, \alpha^-\}$.
(Rel) $E_0 \to^{\mu} E_1$ implies $E_0[\phi] \to^{\phi(\mu)} E_1[\phi]$.
(Sum) $E_0 \to^{\mu} E_1$ implies $E_0 + E \to^{\mu} E_1$ and $E + E_0 \to^{\mu} E_1$.
(Com) $E_0 \to^{\mu} E_1$ implies $E_0 \mid E \to^{\mu} E_1 \mid E$ and $E \mid E_0 \to^{\mu} E \mid E_1$;
$E_0 \to^{\lambda} E_1$ and $E'_0 \to^{\lambda^-} E'_1$ implies $E_0 \mid E'_0 \to^{\tau} E_1 \mid E'_1$.
(Rec) $E_0[\mathrm{rec}\, x.\, E_0/x] \to^{\mu} E_1$ implies $\mathrm{rec}\, x.\, E_0 \to^{\mu} E_1$.

Hereto, we will use the following conventions to talk about sequences of actions and sequences of visible actions:

- $E \Rightarrow^{\varepsilon} E'$, ε being the null string of Λ^*, stands for $E \rightarrow^{\tau^n} E'$, $n \geq 0$;
- $E \Rightarrow^{\mu} E'$, stands for there exist E_1 and E_2 such that $E \Rightarrow^{\varepsilon} E_1 \rightarrow^{\mu} E_2 \Rightarrow^{\varepsilon} E'$;
- $E \Rightarrow^{s} E'$, $s = \lambda_1 \ldots \lambda_n \in \Lambda^+$, stands for there exist E_i, $0 < i < n$, such that $E = E_0 \Rightarrow^{\lambda_1} E_1 \Rightarrow^{\lambda_2} \cdots \Rightarrow^{\lambda_n} E_n = E'$;
- the relation $\Rightarrow^{s}$, $s \in \Lambda^*$, will be referred to as *many step derivation.*

The derivation relation of Definition 2.2 completely specifies the operational semantics of CCS; the second level of CCS semantics is obtained by abstracting from unwanted details. To this purpose, a notion of bisimulation is introduced which is then used to define an equivalence relation on agents. Agents which are observationally equivalent can then be identified.

We can define a bisimulation relation $\boldsymbol{R}$ between CCS agents which consists of all those pairs of agents related via $\Rightarrow^{s}$ to equal (up to $\boldsymbol{R}$) agents. Loosely speaking, two agents E_0 and E_1 are considered as equivalent, written $E_0 \approx E_1$, if and only if there exists a *bisimulation* $\boldsymbol{R}$ containing the pair $\langle E_0, E_1\rangle$ and guaranteeing that E_0 and E_1 are able to perform equal sequences of visible actions evolving to equal (up to $\boldsymbol{R}$) agents.

Definition 2.3 (*observational equivalence*). (1) If $\boldsymbol{R}$ is a binary relation between CCS agents, then Ψ, a function from relations to relations, is defined as follows: $\langle E_0, E_1\rangle \in \Psi(\boldsymbol{R})$ if, for every $s \in \Lambda^*$,

(i) whenever $E_0 \Rightarrow^{s} E_0'$ there exists E_1' such that $E_1 \Rightarrow^{s} E_1'$ and $\langle E_0', E_1'\rangle \in \boldsymbol{R}$,

(ii) whenever $E_1 \Rightarrow^{s} E_1'$ there exists E_0' such that $E_0 \Rightarrow^{s} E_0'$ and $\langle E_0', E_1'\rangle \in \boldsymbol{R}$.

(2) A relation $\boldsymbol{R}$ is a *bisimulation* if $\boldsymbol{R} \subseteq \Psi(\boldsymbol{R})$.

(3) Relation $\approx\, = \bigcup \{\boldsymbol{R} \mid \boldsymbol{R} \subseteq \Psi(\boldsymbol{R})\}$, is called *observational equivalence.*

Proposition 2.4

- *Function Ψ is monotonic on the lattice of relations under inclusion.*
- *Relation $\approx$ is a bisimulation and an equivalence relation.*

Below we present two pairs of equivalent processes. The first shows that the equivalence based on bisimulation succeeds in ignoring the internal structure of agents; the second shows that concurrent and nondeterministic processes may be identified.

Example 2.5. (a) $\alpha.(\beta.\mathrm{NIL} + \tau.\gamma.\mathrm{NIL}) + \alpha.\gamma.\mathrm{NIL} \approx \alpha.(\beta.\mathrm{NIL} + \tau.\gamma.\mathrm{NIL})$;

(b) $\alpha.\mathrm{NIL} \mid \beta.\mathrm{NIL} \approx \alpha.\beta.\mathrm{NIL} + \beta.\alpha.\mathrm{NIL}$.

Here, the relevant bisimulations are

(a) $\{\langle \alpha.(\beta.\mathrm{NIL} + \tau.\gamma.\mathrm{NIL}) + \alpha.\gamma.\mathrm{NIL},\ \alpha.(\beta.\mathrm{NIL} + \tau.\gamma.\mathrm{NIL})\rangle,$
$\langle \beta.\mathrm{NIL} + \tau.\gamma.\mathrm{NIL},\ \beta.\mathrm{NIL} + \tau.\gamma.\mathrm{NIL}\rangle, \langle \gamma.\mathrm{NIL}, \gamma.\mathrm{NIL}\rangle, \langle \mathrm{NIL}, \mathrm{NIL}\rangle\}$,

(b) $\{\langle \alpha.\mathrm{NIL} \mid \beta.\mathrm{NIL},\ \alpha.\beta.\mathrm{NIL} + \beta.\alpha.\mathrm{NIL}\rangle, \langle \mathrm{NIL} \mid \beta.\mathrm{NIL}, \beta.\mathrm{NIL}\rangle,$
$\langle \alpha.\mathrm{NIL} \mid \mathrm{NIL}, \alpha.\mathrm{NIL}\rangle, \langle \mathrm{NIL} \mid \mathrm{NIL}, \mathrm{NIL}\rangle\}$.

Rather than equivalence relations we need congruences which guarantee that equivalent agents can be interchangeably plugged into any context, without affecting the overall behaviour. It is well known that observational equivalence is not preserved by +-contexts, and thus in [27] and [29] this relation is strengthened to a congruence. The definition below characterizes observational congruence without making any explicit use of contexts.

Definition 2.6 (*observational congruence*, Milner [29]). $E_0 \approx^c E_1$ iff
(i) whenever $E_0 \rightarrow^{\mu} E_0'$, there exists E_1' such that $E_1 \Rightarrow^{\mu} E_1'$ and $E_0' \approx E_1'$,
(ii) whenever $E_1 \rightarrow^{\mu} E_1'$, there exists E_0' such that $E_0 \Rightarrow^{\mu} E_0'$ and $E_0' \approx E_1'$.

This definition shows exactly in what respect observational congruence differs from observational equivalence; however, it has the disadvantage of needing explicit concatenations of visible and invisible actions. We aim at giving a similar definition of congruence where partial orderings of events are considered instead of interleavings of events; while string concatenation is trivial, problems arise when a general notion of concatenation on partial orderings is needed. Thus, we introduce below a less elegant characterization of observational congruence, the pattern of which will be followed in defining the partial ordering one in Section 4. The alternative congruence again uses observational equivalence, but takes into account only non-empty initial sequences of silent actions.

Definition 2.7 (*another characterization of observational congruence*). $E_0 \cong^c E_1$ iff
(i) $E_0 \approx E_1$,
(ii) whenever $E_0 \Rightarrow^{\tau} E_0'$, there exists E_1' such that $E_1 \Rightarrow^{\tau} E_1'$ and $E_0' \approx E_1'$,
(iii) whenever $E_1 \Rightarrow^{\tau} E_1'$, there exists E_0' such that $E_0 \Rightarrow^{\tau} E_0'$ and $E_0' \approx E_1'$.

Proposition 2.8 (the two context independent equivalences are the same congruence). $E_0 \approx^c E_1$ *if and only if* $E_0 \cong^c E_1$.

Proof. Immediate, by definition of observational equivalence and by noticing that $E_0 \rightarrow^{\mu} E_0'$ implies $E_0 \Rightarrow^{\mu} E_0'$, and that $E_0 \Rightarrow^{\tau} E_0'$ if and only if $E_0 \rightarrow^{\tau} E \Rightarrow^{\varepsilon} E_0'$, □

3. Defining the partial ordering derivation relation

In this section we define the partial ordering derivation relation $I_1 \rightarrow^{[\mu, \mathscr{R}]} I_2$, which generalizes Milner's derivation relation $E_1 \rightarrow^{\mu} E_2$, and allows us to obtain a notion of many step derivation based on partial orderings.

We first need to single out those subagents of a given CCS agent which can be considered as single entities, in that they may perform actions in isolation.

Definition 3.1 (*defining CCS sequential agents*). A *grape* is a term defined by the following BNF-like grammar

$$G ::= \text{NIL} \mid \mu.E \mid E+E \mid \text{rec } x.\ E \mid \text{id} \mid G \mid G \mid \text{id} \mid G \backslash \alpha \mid G[\phi]$$

where E, $\backslash\alpha$, $[\phi]$ have the standard CCS meaning.

Intuitively speaking, a grape represents either a sequential agent (expressed by the first four alternatives, discussed after Definition 3.2) or a subagent of a CCS agent, together with its access path. The latter is used to take into account the context in which sequential processes operate. This information is crucial on many occasions. For example, it allows us to differentiate the behaviour of processes like $(\alpha.\beta.\text{NIL} \mid \alpha^{-}.\text{NIL})\backslash\alpha$ and $(\alpha.\beta.\text{NIL})\backslash\alpha \mid (\alpha^{-}.\text{NIL})\backslash\alpha$. We have an operator on grapes for each CCS operator and we keep the same name for all the operators apart from that for parallel composition. This is replaced by two unary operators, |id and id|, which are tags recording that there are other processes that can perform actions concurrently with those of the given sequential process.

A CCS agent is decomposed by function dec into a set of grapes.

Definition 3.2 (*decomposing a CCS agent into its sequential agents*). Function dec decomposes a CCS agent into a set of grapes and is defined by structural induction as follows:

$$\text{dec}(\text{NIL}) = \{\text{NIL}\}, \qquad \text{dec}(\mu.E) = \{\mu.E\},$$

$$\text{dec}(E\backslash\alpha) = \text{dec}(E)\backslash\alpha, \qquad \text{dec}(E[\phi]) = \text{dec}(E)[\phi],$$

$$\text{dec}(E_1 + E_2) = \{E_1 + E_2\}, \qquad \text{dec}(E_1 \mid E_2) = \text{dec}(E_1) \mid \text{id} \cup \text{id} \mid \text{dec}(E_2),$$

$$\text{dec}(\text{rec } x.\ E) = \{\text{rec } x.\ E\}.$$

In this definition, and from now onwards, the application of a syntactic constructor to a set of grapes is defined as applying the constructor elementwise, e.g., $I\backslash\alpha = \{g\backslash\alpha \mid g \in I\}$.

The decomposition goes inside the structure of agents and stops when a process prefixed by an action or the NIL process are encountered, since these cannot be considered but atomic sequential processes. It also stops when a sum or a recursion is encountered; this choice is debatable. For example, if we take agent $\alpha.\text{NIL} \mid \beta.\text{NIL} + \gamma.\text{NIL}$ it is not immediate whether it should be considered as a single sequential process, or rather as two sequential processes, namely $\alpha.\text{NIL} \mid \text{id} + \gamma.\text{NIL}$ and $\text{id} \mid \beta.\text{NIL} + \gamma.\text{NIL}$. We take here the first standing and assume that, in order to resolve the choice between the two sides of a +, all concurrent processes on the same side must agree on being chosen. A similar situation arises with recursively defined agents, where all concurrent agents in the rec body must unwind at the same time.

The above assumption of *centralized* control contrasts with that of [11]. There, a decomposition relation decrel is defined which does not consider as sequential those agents having + and rec as top-level operators, and goes always inside the structure of agents. In the case of +, this results in a cartesian product of the sequential components of the alternative agents, thus yielding a combinatorial explosion of the number of generated grapes, and the loss of the one-to-one correspondence between states and CCS agents. Indeed, the alternatives present in all grapes are discarded by the occurrence of a transition only in those grapes affected by it. Nevertheless, the alternatives still present in the remaining grapes are meaningless and will never be taken. Decomposing the above agent $\alpha.\mathrm{NIL}|\beta.\mathrm{NIL}+\gamma.\mathrm{NIL}$ through decrel results in the set of grapes $\{\alpha.\mathrm{NIL}|\mathrm{id}+\gamma.\mathrm{NIL},\ \mathrm{id}|\beta.\mathrm{NIL}+\gamma.\mathrm{NIL}\}$. When the action α is performed, state $\{\mathrm{NIL}|\mathrm{id},\ \mathrm{id}|\beta.\mathrm{NIL}+\gamma.\mathrm{NIL}\}$ is reached where the $\gamma.\mathrm{NIL}$ choice is still present, yet useless.

Example 3.3

$$\begin{aligned}&\mathrm{dec}((((\mathrm{rec}\ x.\ \alpha.x+\beta.x)\,|\,\mathrm{rec}\ x.\ \alpha.x+\gamma.x)\,|\,\mathrm{rec}\ x.\ \alpha^{-}.x)\backslash\alpha)\\&\quad=\{(((\mathrm{rec}\ x.\ \alpha.x+\beta.x)\,|\,\mathrm{id})\,|\,\mathrm{id})\backslash\alpha,\ ((\mathrm{id}\,|\,\mathrm{rec}\ x.\ \alpha.x+\gamma.x)\,|\,\mathrm{id})\backslash\alpha,\\&\qquad(\mathrm{id}\,|\,\mathrm{rec}\ x.\ \alpha^{-}.x)\backslash\alpha\}.\end{aligned}$$

Example 3.4

$$\begin{aligned}&\mathrm{dec}(((\alpha.\mathrm{NIL}\,|\,\gamma.\mathrm{NIL}+\theta.\mathrm{NIL})\,|\,(\alpha^{-}.\mathrm{NIL}\,|\,\delta.\mathrm{NIL}+\upsilon.\mathrm{NIL}))\,|\,\beta.\mathrm{NIL})\\&\quad=\{((\alpha.\mathrm{NIL}\,|\,\gamma.\mathrm{NIL}+\theta.\mathrm{NIL})\,|\,\mathrm{id})\,|\,\mathrm{id},\\&\qquad(\mathrm{id}\,|\,(\alpha^{-}.\mathrm{NIL}\,|\,\delta.\mathrm{NIL}+\upsilon.\mathrm{NIL}))\,|\,\mathrm{id},\ \mathrm{id}\,|\,\beta.\mathrm{NIL}\}.\end{aligned}$$

We now define a correspondence between CCS agents and sets of grapes, more precisely with the sets of their sequential subagents.

Definition 3.5. A set of grapes I is *complete* if there exists a CCS agent E such that $\mathrm{dec}(E)=I$.

Full information about a CCS agent E is retained in $\mathrm{dec}(E)$, since the following property holds.

Property 3.6. *Function* dec *is injective and thus defines a bijection between CCS agents and complete sets of grapes.*

Proof. Immediate by induction. □

Note that the inverse function of dec is standard unification, provided that distinct variables are substituted for each occurrence of id, and $\{\mu E\}$, $\{E_1+E_2\}$ and $\{\mathrm{rec}\ x.\ E\}$ are considered atomic. In other words, the unique unifier of a complete set of grapes is the CCS agent of which it is the decomposition; if a set is contained in a complete set, its unifier is not unique; otherwise there is no unifier.

Sets of grapes will play the role of states in our partial ordering derivations, which are defined below. First, we need some notation used to describe the causal relation between sets of grapes.

Notation 3.7. Let $\mathcal{R}$ be a binary relation, by $\mathcal{R}\downarrow 1$ we understand the set $\{x \mid \exists y$ such that $\langle x, y\rangle \in \mathcal{R}\}$ and by $\mathcal{R}\downarrow 2$ the set $\{y \mid \exists x$ such that $\langle x, y\rangle \in \mathcal{R}\}$. Sometimes we will write $x \leq y$ if $\langle x, y\rangle \in \mathcal{R}$; moreover we will define a relation $\mathcal{R}$ by writing all and only its pairs. Furthermore, we also consider operators to be extended to $\mathcal{R}$, e.g.,

$$\mathcal{R} \mid \mathrm{id} = \{\langle x \mid \mathrm{id}, y \mid \mathrm{id}\rangle, \mid \langle x, y\rangle \in \mathcal{R}\}.$$

The partial ordering derivation relation $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$ is defined via axioms and inference rules in direct correspondence with those of Milner's $E_1 \rightarrow^{\mu} E_2$. In this new relation, sets of grapes (I_1 and I_2), rather than agents, are source and target of the arrow, and $\mathcal{R}$ is a binary relation on grapes. Still, the intuitive meaning of $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$ is that I_1 may become I_2 by performing action μ; thus, we say that the grapes of I_1 *cause* through μ those in I_2 (also written as $I_1 \leq \mu \leq I_2$). The information about other grapes which can be caused by I_1 but not by μ is recorded in $\mathcal{R}$. More precisely, if $g_1 \leq g_2 \in \mathcal{R}$, we have that $g_1 \in I_1$, $g_2 \notin I_2$ and that g_1, but *not* action μ, causes grape g_2. As a whole, we may say that the derivation $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$ replaces the grapes of I_1 with those of $I_2 \cup \mathcal{R}\downarrow 2$ while showing μ. Thus, $\mathcal{R}$ records that there are agents that may perform actions concurrently with μ.

In order to make examples more readable, we now resort to a graphical representation of $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$, already informally used in the introductory section. The causal relation is represented through its Hasse diagram growing downwards (the lines representing the transitive closure are omitted), and since sets I_1 and $I_2 \cup \mathcal{R}\downarrow 2$ may be intersecting, distinct instances of grapes in I_1, I_2 and $\mathcal{R}\downarrow 2$ are considered. The derivation $\{\mathrm{rec}\ x.\ \alpha x\} \rightarrow^{[\alpha,\emptyset]} \{\mathrm{rec}\ x.\ \alpha x\}$ is shown in Fig. 3a; notice that two instances of the same grape are depicted. A formal account of the instantiation construction is given in Definition 4.3. The derivation

$$\{g_1, g_2\} \xrightarrow{[\tau,\{g_1 \leq g_4, g_2 \leq g_5\}]} \{g_6, g_7\}$$

is shown in Fig. 3b; for a denotation of the g_is, see Example 3.10.

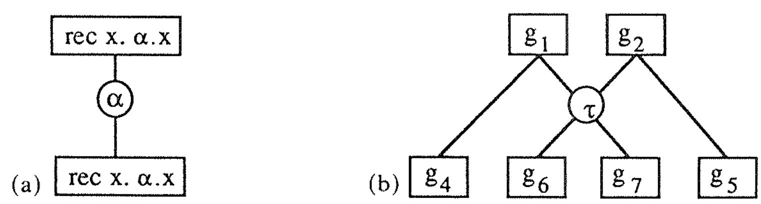


Fig. 3. The graphical representation of $\{\mathrm{rec}\ x.\ \alpha x\} \rightarrow^{[\alpha,\emptyset]} \{\mathrm{rec}\ x.\ \alpha x\}$ (in a), and of $\{g_1, g_2\} \rightarrow^{[\tau,\{g_1 \leq g_4, g_2 \leq g_5\}]} \{g_6, g_7\}$ (in b).

Definition 3.8 (*partial ordering derivation relation*). The *partial ordering derivation relation* $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$ is defined as the least relation satisfying the following axiom and inference rules

(act) $\{\mu E\} \rightarrow^{[\mu,\emptyset]} \text{dec}(E)$,

(res) $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$ implies $I_1 \backslash \alpha \rightarrow^{[\mu,\mathcal{R}\backslash\alpha]} I_2 \backslash \alpha$ if $\mu \notin \{\alpha, \alpha^-\}$,

(rel) $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_1$ implies $I_1[\phi] \rightarrow^{[\phi(\mu),\mathcal{R}[\phi]]} I_2[\phi]$,

(sum) $(\text{dec}(E_1) - I_3) \rightarrow^{[\mu,\mathcal{R}]} I_2$ implies

$$\{E + E_1\} \xrightarrow{[\mu,\{E+E_1\} \leq (I_3 \cup \mathcal{R}\downarrow 2)]} I_2 \quad \text{and}$$

$$\{E_1 + E\} \xrightarrow{[\mu,\{E_1+E\} \leq (I_3 \cup \mathcal{R}\downarrow 2)]} I_2,$$

(com) $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$ implies $I_1|\text{id} \rightarrow^{[\mu,\mathcal{R}|\text{id}]} I_2|\text{id}$ and $\text{id}|I_1 \rightarrow^{[\mu,\text{id}|\mathcal{R}]} \text{id}|I_2$,
$I_1 \rightarrow^{[\lambda,\mathcal{R}]} I_2$ and $I'_1 \rightarrow^{[\lambda^-,\mathcal{R}']} I'_2$ implies

$$I_1|\text{id} \cup \text{id}|I'_1 \xrightarrow{[\tau,\mathcal{R}|\text{id} \cup \text{id}|\mathcal{R}']} I_2|\text{id} \cup \text{id}|I'_2,$$

(rec) $(\text{dec}(E[\text{rec}\, x.\, E/x]) - I_3) \rightarrow^{[\mu,\mathcal{R}]} I_2$ implies

$$\{\text{rec}\, x.\, E\} \xrightarrow{[\mu,\{\text{rec}\, x.\, E\} \leq (I_3 \cup \mathcal{R}\downarrow 2)]} I_2.$$

We can now briefly comment on our axiom and rules. In axiom (act), a single grape is rewritten as a set of grapes, since the firing of the action makes explicit the (possible) parallelism of E. As every grape in $\text{dec}(E)$ is caused by μ, obviously relation $\mathcal{R}$ is empty. Rules (res) and (rel) and the first two rules for (com) simply state that if a set of grapes I_1 can be rewritten as I_2 via μ, then we can combine the access paths of the grapes in both sets with either path constructors $.\backslash\alpha$, $.[\phi]$, $.|\text{id}$ or $\text{id}|.$, and still obtain a derivation, labelled, say, by μ'. When dealing with restriction we have that μ' is μ, but the inference is possible only if $\mu \notin \{\alpha, \alpha^-\}$; in (rel) μ' is $\phi(\mu)$ and in the first two rules of (com) μ' is simply μ. Relation $\mathcal{R}$ is accordingly modified. The third rule for (com) is the synchronization rule; of course it takes care that relations $\mathcal{R}$ and $\mathcal{R}'$ are (modified and) unioned.

A derivation generated by the first implication of rule (sum) can be understood as consisting of two steps. Starting from the singleton $\{E_1 + E\}$ the first step discards alternative E and decomposes E_1 into the union of suitable sets of grapes I_1 and I_3; the second step (the premise of the inference rule) rewrites I_1 as I_2, leaving I_3 idle (see also Fig. 2a). The grapes in I_3 are originated by $E_1 + E$ but not caused by μ, so we add $\{E_1 + E\} \leq I_3$ to $\mathcal{R}$. Moreover, all the grapes which are caused by some grape in $\text{dec}(E_1) - I_3$ and not by μ (namely, those in $\mathcal{R}\downarrow 2$) are still caused by $\{E_1 + E\}$ and not by μ, thus we also have that $\{E_1 + E\} \leq \mathcal{R}\downarrow 2$ is in $\mathcal{R}$. The net effect of these two steps is then rewriting the singleton $\{E_1 + E\}$ into the set I_2 and labelling the arrow with the pair $[\mu, \{E_1 + E\} \leq (I_3 \cup \mathcal{R}\downarrow 2)]$. Similarly for the second rule of (sum).

The intuition behind rule (rec) is similar to that behind (sum). Obviously, the first step consists now in unwinding the recursive agent in *all* the occurrences of the bound variable, rather than discarding one of the alternatives.

The way we deal with nondeterministic choice and recursion shows that our transition rules still assume a centralized control. For instance, all the concurrent sequential processes which occur in an argument of + must participate in and are affected by the decision.

The following property clarifies the structure of the derivations and stresses the asynchrony of the partial ordering derivation relation. Indeed, the underlying model of standard CCS derivations is a transition system, while Definition 3.8 introduces a rewriting system. $E_1 \rightarrow^{\mu} E_2$ is a transition, i.e., E_1 and E_2 are global states, while $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$ can be interpreted as a rewriting rule, since the grapes involved there are only those processes of the current state which are active in the step. The correspondence between the two derivation relations is stated in Theorem 3.11.

Property 3.9. *Given* $I_1 \rightarrow^{[\mu,\mathcal{R}]} I_2$ *in the partial ordering derivation relation, we have*
- $\mathcal{R}\downarrow 1 \subseteq I_1$,
- $\mathcal{R}\downarrow 2 \cap I_2 = \emptyset$,
- *for every set of grapes* I, $I_1 \cup I$ *is complete if and only if* $I_2 \cup \mathcal{R}\downarrow 2 \cup I$ *is complete.*

Proof. Immediate by induction. □

Example 3.10. Let us consider the agent of Example 3.4.

$$E_1 = ((\alpha.\text{NIL} \mid \gamma.\text{NIL} + \theta.\text{NIL}) \mid (\alpha^{-}.\text{NIL} \mid \delta.\text{NIL} + \upsilon.\text{NIL})) \mid \beta.\text{NIL}$$

and the agent $E = \eta.\text{NIL}$. Furthermore, let

$$g_1 = ((\alpha.\text{NIL} \mid \gamma.\text{NIL} + \theta.\text{NIL}) \mid \text{id}) \mid \text{id},$$

$$g_2 = (\text{id} \mid (\alpha^{-}.\text{NIL} \mid \delta.\text{NIL} + \upsilon.\text{NIL})) \mid \text{id},$$

$$g_3 = \text{id} \mid \beta.\text{NIL},$$

$$g_4 = ((\text{id} \mid \gamma.\text{NIL}) \mid \text{id}) \mid \text{id},$$

$$g_5 = (\text{if} \mid (\text{id} \mid \delta.\text{NIL})) \mid \text{id},$$

$$g_6 = ((\text{NIL} \mid \text{id}) \mid \text{id}) \mid \text{id},$$

$$g_7 = (\text{id} \mid (\text{NIL} \mid \text{id})) \mid \text{id},$$

$$\mathcal{R} = \{g_1 \leqslant g_4, g_2 \leqslant g_5\}.$$

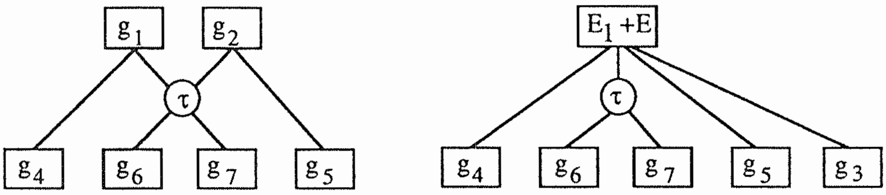
By using the first inference rule (sum), from the partial ordering derivation

$$\{g_1, g_2\} \rightarrow^{[\tau,\mathcal{R}]} \{g_6, g_7\},$$

shown in Fig. 4a, we can deduce the derivation of Fig. 4b

$$\{E_1 + E\} \xrightarrow{[\tau,\{E_1+E\} \leqslant (\{g_3\} \cup \mathcal{R}\downarrow 2)]} \{g_6, g_7\}.$$

In fact, we have that $\text{dec}(E_1) = \{g_1, g_2, g_3\}$ (see Example 3.4), thus I_3 contains g_3, and $\mathcal{R}\downarrow 2 = \{g_4, g_5\}$.



a) **b)**

Fig. 4. The partial ordering derivations $\mathrm{dec}(E_1)-\{g_3\}=\{g_1, g_2\}\rightarrow^{[\tau,\{g_1\leqslant g_4, g_2\leqslant g_5\}]}\{g_6, g_7\}$ (in a), and $\{E_1+E\}\rightarrow^{[\tau,\{E_1+E\}\leqslant\{g_3, g_4, g_5\}]}\{g_6, g_7\}$ (in b).

Theorem 3.11 (correspondence between Milner's and partial ordering derivations). *We have a derivation* $E_0\rightarrow^{\mu} E_1$ *if and only if there exist a relation* $\mathcal{R}$ *and a set of grapes* I *such that* $(\mathrm{dec}(E_0)-I)\rightarrow^{[\mu,\mathcal{R}]}(\mathrm{dec}(E_1)-(\mathcal{R}\downarrow 2\cup I))$.

Proof. Given a derivation of either kind, use the structure of its proof to obtain the derivation of the other kind. □

4. Partial ordering many step derivations and equivalences

In this section we concatenate the derivations given in Section 3 to define computations from which the partial ordering many step derivations for CCS are obtained. The partial orderings of events of these derivations express the complete causal dependencies among the performed events. In order to relate our many step derivations with Milner's, we also introduce total orderings on events that reflect the temporal relation among them. Finally, the two relations of partial ordering observational equivalence and congruence are defined which are based on bisimulation and on the previously given many step derivations.

The next definition introduces three orderings of events which will be used to capture the relevant information about behaviours of agents.

Definition 4.1 (*orderings of events*). Let A be a countable set of *event labels.*

(i) A *partial ordering of events* is a triple $h=\langle S, l, \leqslant\rangle$, where

- S is a finite set of *events*;
- $l: S\rightarrow A$ is a *labelling function*;
- $\leqslant$ is a partial ordering relation on S, called *causal relation.*

(ii) A *total ordering of events* is a partial ordering of events $t=\langle s, l, \lesssim\rangle$ such that $\lesssim$ is total, called *temporal relation.*

(iii) A *mixed ordering of events* is a quadruple $d=\langle S, l, \leqslant, \lesssim\rangle$, where $\langle S, l, \leqslant\rangle$ is a partial ordering and $\langle S, l, \lesssim\rangle$ is a total ordering of events.

Two events e_1 and e_2 are *concurrent* if neither $e_1\leqslant e_2$ nor $e_2\leqslant e_1$. Two orderings of events will be identified if *isomorphic*, i.e., if there is a label- and order-preserving bijection between their events. A total ordering of events (defined up to isomorphism) will be identified with the sequence of the labels of its events. We will define an ordering of events by explicitly writing all and only its pairs.

Figure 5 shows a partial ordering of events, with the conventions that events are represented by circles with their labels inside, and that the partial ordering $\leqslant$ is represented by its Hasse diagram growing downwards. So, we have that event e_1, labelled by α, has no relation with all the others, thus it is concurrent with them all. The event labelled by γ dominates, i.e., *causes* the remaining events. Note that the labelling function is not injective.

We will now introduce our notion of computation, defined as a finite sequence of complete sets of grapes and of partial ordering derivations.

Definition 4.2 (*computation*). A sequence

$$\xi = \{G_0 I_1 \xrightarrow{[\mu_1, \mathcal{R}_1]} I'_1\ G_1 \cdots G_{n-1}\ I_n \xrightarrow{[\mu_n, \mathcal{R}_n]} I'_n G_n\}$$

is a *computation* if

(i) G_i is a complete set of grapes, $0 \leqslant i \leqslant n$, and $I_i \rightarrow^{[\mu_i, \mathcal{R}_i]} I'_i$ is in the partial ordering derivation relation, $0 < i \leqslant n$;

(ii) $I_i \subseteq G_{i-1}$, and $G_i = (G_{i-1} - I_i) \cup \mathcal{R}_i{\downarrow}2 \cup I'_i$, $0 < i \leqslant n$.

As noted in the previous section, the elements of the partial ordering derivation relation are rewriting rules which are applied in the computation above. States are (represented as) complete sets of grapes. This is essentially due to our assumption of having a centralized control. Indeed, function dec induces and Theorem 3.11 establishes this natural one-to-one correspondence between the states of the original interleaving and of the partial ordering computations. In the ith step, state G_{i-1} evolves to G_i by applying $I_i \rightarrow^{[\mu_i, \mathcal{R}_i]} I'_i$ in such a way that the set of grapes I_i (contained in G_{i-1}) is rewritten as $\mathcal{R}_i{\downarrow}2 \cup I'_i$, while the grapes in $G_{i-1} - I_i = G_i - (\mathcal{R}_i{\downarrow}2 \cap I'_i)$ stay idle.

Note also that our notion of computation coincides with Milner's, when a single step is performed, because of the correspondence between his and our derivation relation established by Theorem 3.11.

From computations we generate a mixed ordering of events recording all their temporal and causal dependences. This ordering is obtained in three steps; first an

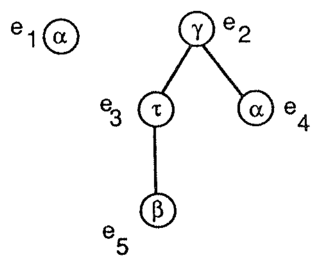


Fig. 5. A partial ordering of events.

event is associated to every derivation and different instances for all the grapes in the global states are created. Then, the causal dependencies among events and grape instances are determined. Finally, all grape instances and all events labelled by τ are removed to obtain the required ordering. The total ordering is determined at once by the ordering in which the rewriting rules are applied during the computation.

Definition 4.3 (*mixed many step derivation*). Given two CCS agents E_0 and E_1, we define the *mixed derivation*, written $E_0 \Rightarrow^d E_1$, iff there exists a computation

$$\xi = \{G_0\ I_1 \xrightarrow{[\mu_1, \mathscr{R}_1]} I'_1\ G_1 \cdots G_{n-1}\ I_n \xrightarrow{[\mu_n, \mathscr{R}_n]} I'_n\ G_n\}$$

where $G_0 = \mathrm{dec}(E_0)$, $G_n = \mathrm{dec}(E_1)$, and $d = \langle S, l, \leqslant, \lesssim\rangle$ is the mixed ordering of events labelled on Λ defined as follows

(i) let $S' = \{e_1, \ldots, e_n\}$ and $B = \{\langle g, i\rangle \mid g \in G_i\}$;

(ii) Let F^* be the reflexive and transitive closure of the flow relation F defined on $S' \cup B$ by the following inference rules

- $g \in \{G_{i-1} - I_i\}$ implies $\langle g, i-1\rangle\ F\ \langle g, i\rangle$,
- $g \in I_i$ implies $\langle g, i-1\rangle\ F\ e_i$,
- $g \in I'_i$ implies $e_i\ F\ \langle g, i\rangle$,
- $\langle g_1, g_2\rangle \in \mathscr{R}_i$ implies $\langle g_1, i-1\rangle\ F\ \langle g_2, i\rangle$;

(iii) $S = \{e_i \mid \mu_i \neq \tau\}$, $l(e_i) = \mu_i$, $\leqslant$ is the restriction of F^* to S, and $e_i \lesssim e_j$, $1 \leqslant i \leqslant j \leqslant n$.

A mixed derivation contains complete information about the evolution of agents. In particular, it records the initial and final agents, the performed events, their generation ordering (expressed through $\lesssim$), and their causal dependencies (expressed through $\leqslant$). This information is extracted from a computation ξ by constructing two sets, the first consisting of events, the second of instances of grapes, and then by determining the orderings over them. Index i in $\langle g, i\rangle$ is used to create a fresh instance of the grape g which occurs in G_i. The link between event e_i and the ith step of the computation $I_i \rightarrow^{[\mu_i, \mathscr{R}_i]} I'_i$ is crucial for determining the orderings. Indeed, the indexes of events are used to recover at once the temporal ordering. The causal relation between grape instances and events is obtained as the reflexive and transitive closure of the causal dependencies expressed by the derivations. More precisely, the first inference rule relates the two instances $\langle g, i-1\rangle$ and $\langle g, i\rangle$ of the grape g, which is idle in the ith step of the computation since it belongs to both $G_{i-1} - I_i$ and $G_i - (\mathscr{R}_i{\downarrow}2 \cup I'_i)$. The second rule makes a grape instance greater than the actual event it performs; the third one is symmetrical. The last rule sets the causal dependencies between a grape instance and those grapes generated by it, but not by the event. Actually, the mixed ordering of events of the mixed derivation is determined by keeping only the events labelled by visible actions, and by restricting the temporal and causal orderings accordingly. Below, Example 4.5 illustrates this construction.

Before giving the example, a further abstraction step is needed; derivations are considered where only the temporal or the causal relations are kept.

Definition 4.4 (*partial ordering and interleaving many step derivations*). Given two CCS agents E_0 and E_1, we have $E_0 \Rightarrow^h E_1$ (called *partial ordering many step derivation*) and $E_0 \Rightarrow^t E_1$ (called *interleaving many step derivation*) iff there exists a mixed derivation $E_0 \Rightarrow^d E_1$, with $d = \langle S, l, \leqslant, \lesssim \rangle$, $h = \langle S, l, \leqslant \rangle$ and $t = \langle S, l, \lesssim \rangle$.

It is worth noting that we build our many step derivations by composing elementary steps and then abstracting. The next example shows the role of relation $\mathcal{R}$ (see the third derivation of the computation) and how causal dependencies are transmitted through τ, and in general how partial ordering of events are obtained from computations.

Example 4.5. Consider the CCS agent

$$E = \alpha.\text{NIL} \mid (\beta^-.\text{NIL} \mid \gamma.((\beta.\beta.\text{NIL} \mid \eta.\text{NIL} + \theta.\text{NIL}) \mid \delta.\text{NIL}));$$

the grapes

$$g_0 = \alpha.\text{NIL} \mid \text{id}, \qquad g_1 = \text{id} \mid (\beta^- \text{NIL} \mid \text{id}),$$

$$g_2 = \text{id} \mid (\text{id} \mid \gamma.((\beta.\beta.\text{NIL} \mid \eta.\text{NIL} + \theta.\text{NIL}) \mid \delta.\text{NIL})), \qquad g_3 = \text{NIL} \mid \text{id},$$

$$g_4 = \text{id} \mid (\text{id} \mid ((\beta.\beta.\text{NIL} \mid \eta.\text{NIL} + \theta.\text{NIL}) \mid \text{id})), \qquad g_5 = \text{id} \mid (\text{id} \mid (\text{id} \mid \delta.\text{NIL})),$$

$$g_6 = \text{id} \mid (\text{NIL} \mid \text{id}), \qquad g_7 = \text{id} \mid (\text{id} \mid ((\beta.\text{NIL} \mid \text{id}) \mid \text{id})),$$

$$g_8 = \text{id} \mid (\text{id} \mid ((\text{id} \mid \eta.\text{NIL}) \mid \text{id})), \qquad g_9 = \text{id} \mid (\text{id} \mid (\text{id} \mid \text{NIL})),$$

$$g_{10} = \text{id} \mid (\text{id} \mid ((\text{NIL} \mid \text{id}) \mid \text{id})), \qquad g_{11} = \text{id} \mid (\text{id} \mid (\text{id} \mid \text{NIL}) \mid \text{id}),$$

and the complete sets of grapes.

$$G_0 = \{g_0, g_1, g_2\}, \qquad G_1 = \{g_3, g_1, g_2\},$$

$$G_2 = \{g_3, g_1, g_4, g_5\}, \qquad G_3 = \{g_3, g_6, g_7, g_8, g_5\},$$

$$G_4 = \{g_3, g_6, g_7, g_8, g_9\}, \qquad G_5 = \{g_3, g_6, g_{10}, g_8, g_9\},$$

$$G_6 = \{g_3, g_6, g_{10}, g_{11}, g_9\}.$$

We have that $G_0 = \text{dec}(E)$, from which the following computation will start.

$$\xi = \{G_0\,\{g_0\} \xrightarrow{[\alpha,\emptyset]} \{g_3\}\ G_1\,\{g_2\} \xrightarrow{[\gamma,\emptyset]} \{g_4, g_5\}\ G_2$$

$$\{g_1, g_4\} \xrightarrow{[\tau,\{g_4 \leq g_8\}]} \{g_6, g_7\}\ G_3\,\{g_5\} \xrightarrow{[\delta,\emptyset]} \{g_9\}\ G_4$$

$$\{g_7\} \xrightarrow{[\beta,\emptyset]} \{g_{10}\}\ G_5\,\{g_8\} \xrightarrow{[\eta,\emptyset]} \{g_{11}\}\ G_6\}.$$

Computation ξ generates the mixed derivation

$$\alpha.\mathrm{NIL}\,|\,(\beta^{-}.\mathrm{NIL}\,|\,\gamma.((\beta.\beta.\mathrm{NIL}\,|\,\eta.\mathrm{NIL}+\theta.\mathrm{NIL})\,|\,\delta.\mathrm{NIL})$$

$$\Rightarrow^{d} \mathrm{NIL}\,|\,(\mathrm{NIL}\,|\,((\mathrm{NIL}\,|\,\mathrm{NIL})\,|\,\mathrm{NIL}))$$

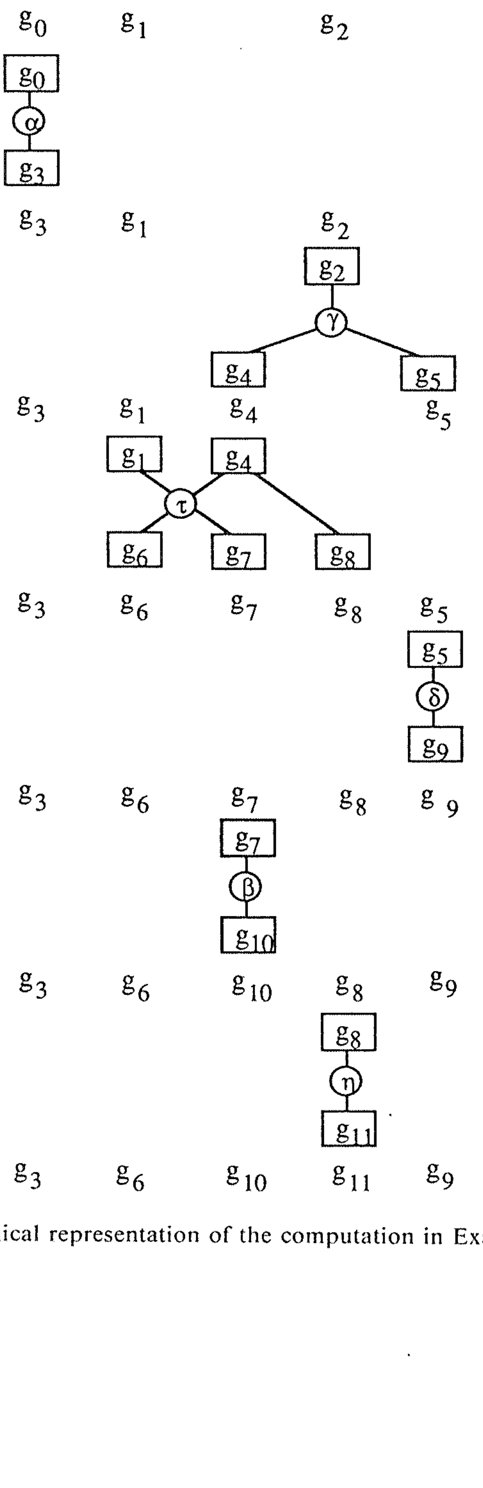


Fig. 6a. A graphical representation of the computation in Example 4.5.

from which the following partial ordering and interleaving many step derivations are extracted: $h=\langle S, l, \leqslant\rangle$ and $t=\langle S, l, \lesssim\rangle$, with

- $S=\{e_1, e_2, e_4, e_5, e_6\}$;
- $l(e_1)=\alpha$, $l(e_2)=\gamma$, $l(e_4)=\delta$, $l(e_5)=\beta$, $l(e_6)=\eta$;
- $e_2 \leqslant e_4$, $e_2 \leqslant e_5$, $e_2 \leqslant e_6$, $e_i \leqslant e_i$,
- $e_1 \lesssim e_2 \lesssim e_4 \lesssim e_5 \lesssim e_6$, $e_i \lesssim e_i$.

Figure 6a depicts the computation ξ; Figure 6b is an intermediate snapshot in getting the partial ordering many step derivation (after having determined the flow relation F). Figure 7 shows h.

We can now state a fundamental theorem about our operational semantics of CCS. The property expressed by Theorem 4.1 (called *complete concurrency* [18])

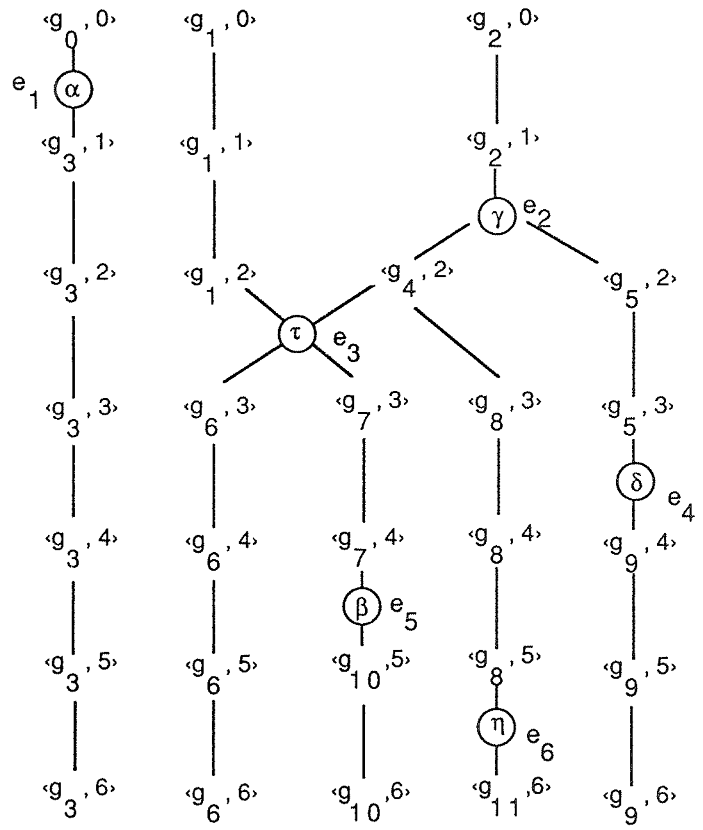


Fig. 6b. The flow relation needed as an intermediate step to obtain its partial ordering of events.

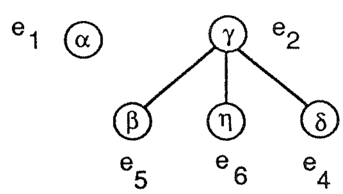


Fig. 7. The partial ordering of the many step derivation of Example 4.5.

relates the total and the partial orderings obtained from computations. More precisely, the first part of the theorem states that, given a computation, the events in the derived mixed ordering of events $\langle S, l, \leqslant, \lesssim\rangle$ are generated in a total temporal ordering that is, as expected, compatible with the causal ordering ($\leqslant \subseteq \lesssim$). The second and crucial part says that these events can be generated by different computations (with the same initial and final set of grapes) in *all* temporal orderings $\lesssim'$ compatible with the causal one ($\leqslant \subseteq \lesssim'$), namely $\leqslant$ is *complete.* Shortly, completeness amounts to saying that any two concurrent events can be generated in either temporal order. As we will see later, complete concurrency plays a crucial role in relating the notions of partial ordering many step derivations with Milner's and therefore in proving that partial ordering observational equivalence and congruence are finer than Milner's.

Theorem 4.6 (complete concurrency). *Given two CCS agents E_0 and E_1 and a mixed ordering of events $d = \langle S, l, \leqslant, \lesssim\rangle$ such that $E_0 \Rightarrow^d E_1$, we have that*

- $\leqslant \subseteq \lesssim$,
- $\forall \lesssim'$ *such that* $\leqslant \subseteq \lesssim'$, *there exists a mixed derivation* $E_0 \Rightarrow^{d'} E_1$, *with* $d' = \langle S, l, \leqslant, \lesssim'\rangle$.

Proof. The proof of the first claim is immediate, the proof of the second one is given in Appendix A. □

The next example shows how unguarded recursion is naturally dealt with in our framework. It also gives evidence that unguardedness may lead to infinitely branching partial orderings that reflect unbounded parallelism.

Example 4.7. Consider the unguarded recursive agent $\text{rec}\,x.\ \alpha.\text{NIL}|x$. It originates the computation

$$\xi = \{\{\text{rec}\,x.\ \alpha.\text{NIL}|x\}$$

$$\text{rec}\,x.\ \alpha.\text{NIL}|x \xrightarrow{[\alpha,\text{rec}\,x.\ \alpha.\text{NIL}|x \leqslant \text{id}|\text{rec.}\,x.\ \alpha.\text{NIL}|x]} \text{NIL}|\text{id}$$

$$\{\text{NIL}|\text{id},\ \text{id}|\text{rec}\,x.\ \alpha.\text{NIL}|x\}$$

$$\text{id}|\text{rec}\,x.\ \alpha.\text{NIL}|x \xrightarrow{[\alpha,\text{id}|\text{rec}\,x.\ \alpha.\text{NIL}|x \leqslant \text{id}|(\text{id}|\text{rec}\,x.\ \alpha.\text{NIL}|x)]} \text{id}|(\text{NIL}|\text{id})$$

$$\{\text{NIL}|\text{id},\ \text{id}|(\text{NIL}|\text{id}),\ \text{id}(\text{id}|\text{rec}\,x.\ \alpha.\text{NIL}|x)\}\}.$$

(Note that, according to Milner, we would have

$$\text{rec}\,x.\ \alpha.\text{NIL}|x \rightarrow^{\alpha} \text{NIL}|\text{rec}\,x.\ \alpha.\text{NIL}|x \rightarrow^{\alpha} \text{NIL}|\text{NIL}|\text{rec}\,x.\ \alpha.\text{NIL}|x.)$$

We have that $\text{rec}\,x.\ \alpha.\text{NIL}|x \Rightarrow^h \text{NIL}|(\text{NIL}|\text{rec}\,x.\ \alpha.\text{NIL}|x)$, where the partial ordering of events h consists of two concurrent events labelled by α. Figure 8 shows an intermediate step in getting h, i.e., the flow relation.

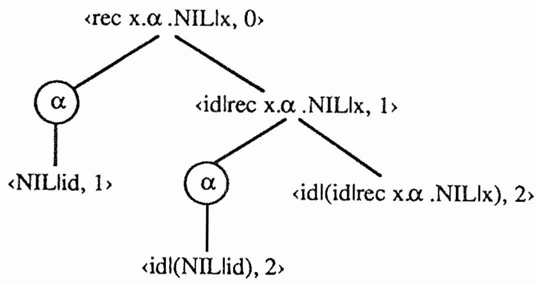


Fig. 8. The flow relation needed as an intermediate step in getting the partial ordering of the many step derivation of Example 4.7.

We have already noted that, due to the assumption of centralized control, there is a natural bijection between the states of the interleaving and of the partial ordering computations: the states of partial ordering computations are all and only complete sets of grapes, i.e., decompositions of CCS agents. The following theorem shows that interleaving many step derivations coincide with Milner's, since total orderings of events are considered as the sequences of the labels of their events.

Theorem 4.8 (deriving Milner's many step derivations from mixed ones). *Given two CCS agents E and E', we have Milner's many step derivation $E \Rrightarrow^s E'$ if and only if there exists an interleaving many step derivation $E \Rrightarrow^s E'$.*

Proof. Let

$$\xi = \{G_0\ I_1 \xrightarrow{[\mu_1,\mathcal{R}_1]} I'_1\ G_1 \cdots G_{n-1}\ I_n \xrightarrow{[\mu_n,\mathcal{R}_n]} I'_n G_n\}$$

be a computation such that $E \Rrightarrow' E'$ holds. We build, by inducing on the length of ξ, a sequence of Milner's derivation rules

$$E = E_0 \xrightarrow{\mu_1} E_1\ E_1 \xrightarrow{\mu_2} E_2 \cdots E_{n-1} \xrightarrow{\mu_n} E_n = E'$$

such that $E \Rrightarrow' E'$.

When there is no step, the claim follows trivially. Assume inductively that the thesis holds at the ith step: we have then that there exists an agent E_{i-1} such that $G_{i-1} = \mathrm{dec}(E_{i-1})$. By Theorem 3.11 and by definition of computation, there exists a set of grapes I_3 such that $G_{i-1} - I_3 = I_i$, $G_i - (\mathcal{R}_i{\downarrow}2 \cup I_3) = I'_i$, and $I_i \to^{[\mu_i,\mathcal{R}_i]} I'_i$ if $E_{i-1} \to^{\mu_i} E_i$, with $\mathrm{dec}(E_i) = G_i$. Since both many step derivations forget τs, the proof of the inductive step follows. The proof of the only if part is symmetric. □

A consequence of Theorems 4.6 and 4.8 is that Milner's many step derivations can be easily recovered from partial ordering many step derivations, the former being just linearizations of the latter. In other words, the original interleaving operational semantics for CCS is immediately retrievable from ours, since there is a syntactical bijection between the two kinds of derivation.

Theorem 4.9 (Milner's many step derivations are interleavings of partial ordering many step derivations). *Given two CCS agents E_0 and E_1,*

(i) *if there exists a partial ordering many step derivation $E_0 \Rightarrow^h E_1$, where $h = \langle S, l, \leqslant \rangle$, then, for all $s = \langle S, l, \lesssim \rangle$ such that $\leqslant \subseteq \lesssim$, we have Milner's many step derivation $E_0 \Rightarrow^s E_1$;*

(ii) *for all Milner's many step derivation $E_0 \Rightarrow^s E_1$, there exists a partial ordering of events $h = \langle S, l, \leqslant \rangle$, with $\leqslant \subseteq \lesssim$ (s is considered as total ordering of events $\langle S, l, \lesssim \rangle$), such that $E_0 \Rightarrow^h E_1$.*

Proof. Theorem 4.8 relates Milner's many step derivations with our interleaving many step derivation. Then, Theorem 4.6 suffices to prove the claim. □

So far, we have abstracted from actual computations to obtain many step derivations by forgetting the intermediate states of computations, the actual temporal ordering in which their events have been generated, and the events labelled by τ. However, we have not yet defined any (semantic) equivalence on CCS agents. Now, we will further abstract from the syntactic structure of agents by defining an equivalence relation over them. The very basic correspondence established by the above theorem makes it possible to carry over the partial ordering approach to the extensional semantics for CCS defined so far (e.g., see [27, 29, 15]). In what follows, we will extend the approach of [29] and define an observational equivalence based on the notion of bisimulation, but this time we rely on partial orderings ($\Rightarrow^h$) rather than on sequences of actions ($\Rightarrow^s$). The following definition thus rephrases Definition 2.3.

Definition 4.10 (*partial ordering observational equivalence*). (1) If $\boldsymbol{R}$ is a binary relation between CCS agents and h is a partial ordering of events, then Θ, a function from relations to relations, is defined as follows: $\langle E_1, E_2 \rangle \in \Theta(\boldsymbol{R})$ if

(i) whenever $E_1 \Rightarrow^h E_1'$ there exists E_2' such that $E_2 \Rightarrow^h E_2'$ and $\langle E_1', E_2' \rangle \in \boldsymbol{R}$,

(ii) whenever $E_2 \Rightarrow^h E_2'$ there exists E_1' such that $E_1 \Rightarrow^h E_1'$ and $\langle E_1', E_2' \rangle \in \boldsymbol{R}$.

(2) A relation $\boldsymbol{R}$ is a *bisimulation* if $\boldsymbol{R} \subseteq \Theta(\boldsymbol{R})$.

(3) Relation $\cong = \bigcup \{\boldsymbol{R} \mid \boldsymbol{R} \subseteq \Theta(\boldsymbol{R})\}$, is called *partial ordering observational equivalence.*

Proposition 4.11.

- *Function Θ is monotonic on the lattice of relations under inclusion.*
- *Relation $\cong$ is a bisimulation and equivalence relation.*

Example 4.12. It is easy to verify, for every agent E, that

(a) the following equivalences hold

(i) $\alpha.E \cong \alpha.\tau.E$,

(ii) $\alpha.E \cong \tau.\alpha.E$,

(iii) $\alpha.(\beta.E'+\tau.\gamma.E)+\alpha.\gamma.E \cong \alpha.(\beta.E'+\tau.\gamma.E)$
(note that (i) and (iii) are two of the τ-laws of [27]);
(b) it is *not true* that $\alpha.\text{NIL}|\beta.\text{NIL} \cong \alpha.\beta.\text{NIL}+\beta.\alpha.\text{NIL}$.

Not suprisingly, the above defined partial ordering observational equivalence is finer than observational equivalence.

Theorem 4.13 (partial ordering equivalence is finer than observational equivalence). *Given two CCS agents E_0 and E_1, we have that $E_0 \cong E_1$ implies $E_0 \approx E_1$, but not vice versa.*

Proof. To show that $\cong$ implies $\approx$ it suffices to prove that, given any bisimulation relation R_{po} based on partial ordering of events and such that $\langle E_1, E_2\rangle \in R_{\text{po}}$, it is possible to define a new relation R_{int} based on total ordering of events and such that $\langle E_1, E_2\rangle \in R_{\text{int}}$ and $R_{\text{int}} \subseteq \Theta(R_{\text{int}})$. This is easily done, since $E \Rrightarrow^h E'$ implies $E \Rrightarrow^t E'$, for all $t \supseteq h$, by Theorem 4.6. Thus, we can choose R_{int} to be R_{po} itself. The claim follows, by applying Theorem 4.8 which establishes the one-to-one correspondence between interleaving many step derivations and Milner's. Example 4.12(b) shows that $E_0 \approx E_1$ does not imply $E_0 \cong E_1$. □

Theorem 4.14. *Partial ordering observational equivalence is preserved by operators* NIL, $\mu.$, $\backslash\alpha$, $[\phi]$ *and* $|$.

Proof. See Appendix B. □

We will now refine the notion of partial ordering observational equivalence so that the new relation is preserved under all contexts. The following definition follows the pattern of the context independent characterization of observational congruence given in Section 2. Again, two agents are congruent if they are equivalent and, whenever one may perform at least one τ, the other may do so as well, becoming equivalent agents. We need a definition first.

Definition 4.15 (*nonempty sequences of silent transitions*). We write $E \Rrightarrow^\tau E'$ if and only if there exists a computation with at least one step involving only partial ordering derivations labelled by τ.

Definition 4.16 (*partial ordering congruence*). Two CCS agents E_0 and E_1 are *partial ordering observational congruent*, written as $E_0 \cong^c E_1$, if and only if

(i) $E_0 \cong E_1$,
(ii) whenever $E_0 \Rrightarrow^\tau E_0'$, there exists an agent E_1' such that $E_1 \Rrightarrow^\tau E_1'$ and $E_0' \cong E_1'$,
(iii) whenever $E_1 \Rrightarrow^\tau E_1'$, there exists an agent E_0' such that $E_0 \Rrightarrow^\tau E_0'$ and $E_0' \cong E_1'$.

Theorem 4.17 ($\cong^c$ is preserved by all contexts). *Relation* $\cong^c$ *is a congruence.*

Proof. Let E, E_0, E_1 be CCS agents. The proof proceeds by case analysis on the operators of CCS, under the hypothesis that there exists a bisimulation relation R containing the pair $\langle E_0, E_1\rangle$ i.e., $E_0 \cong^c E_1$.

The proof in cases (act), (res), (rel) and (com) is immediate since item (i) has been established by Theorem 4.14, and proving items (ii) and (iii) is trivial.

(sum) The only difficult part of proving that $E_0 + E \cong^c E_1 + E$ (and symmetrically that $E + E_0 \cong^c E + E_1$) is showing that $E_0 + E \cong E_1 + E$, i.e., when $E_0 + E \Rrightarrow^h E_0'$ also $E_1 + E \Rrightarrow^h E_1'$ with $E_0' \cong E_1'$; and vice versa. When E_0 moves via visible actions, the proof is trivial. When $E_0 \Rrightarrow^\tau E_0'$, also $E_1 \Rrightarrow^\tau E_1'$, for $E_0 \cong^c E_1$ by hypothesis (in particular, note that item (ii) holds); and vice versa.

(rec) The proof can be carried on by following step by step the corresponding proof for the original interleaving semantics (Proposition 2.7 of [29]). We need extending observational congruence to open terms in order to prove that, given two open terms E_0 and E_1, $E_0 \cong^c E_1$ implies rec x. $E_0 \cong^c$ rec x. E_1. The only difference with the proof of [29] is due to the definition of bisimulation. There, it is based on single-step derivations, while in our case it relies on computations of arbitrary length. Thus, an additional induction on the length of computations is needed. □

As expected, partial ordering congruence is finer than observational congruence; furthermore they concide when dealing with nondeterministic sequential processes only.

Corollary 4.18 (partial ordering congruence is finer than observational congruence). *Given two CCS agents* E_0 *and* E_1, *we have that* $E_0 \cong^c E_1$ *implies* $E_0 \approx^c E_1$, *but not vice versa.*

Proof. The implication follows from Theorems 4.8, 4.9 and 4.13. Example 4.12(b) shows also that the reverse implication does not hold. □

Not surprisingly, the partial ordering equivalence and congruence relations coincide with the original relations introduced in [29] when they are restricted to sequential nondeterministic processes.

Theorem 4.19 (partial ordering congruence and observational congruence coincide on sequential processes). *Let* SEQ *be the set of CCS agents in which* $|$ *does not occur. The restriction of* $\cong^c$ *to* SEQ × SEQ *coincides with the restriction of* $\approx^c$ *to* SEQ × SEQ.

Proof. If E_0 is in SEQ and $E_0 \Rightarrow^s E_1$, Then dec(E_0) is a singleton and such are all the intermediate sets of grapes in the computation. All partial ordering many step derivations $E_0 \Rrightarrow^{\langle S, l, \leqslant\rangle} E_1$ are such that $\leqslant$ is a total ordering of events. Thus, we have that $E_0 \cong^c E_1$ iff $E_0 \approx^c E_1$. □

5. Conclusions and related work

A partial ordering semantics for CCS has been presented which is based on a set of rewriting rules given in the SOS style, and on a notion of observational congruence. A rewriting rule describes the evolution of sets of sequential subagents which are obtained by decomposing CCS agents, and expresses the causal relation among the initial subagents, the performed action, and the resulting subagents. The congruence abstracts from internal behaviour but still distinguishes concurrent execution of actions from their nondeterministic interleavings, and preserves information about the causal relation among them.

To make the choice of a particular true concurrent semantics less arbitrary, in [11] we state two criteria we consider essential to assess any new partial ordering semantics of a language previously equipped with an interleaving one:

(i) the interleaving semantics must be retrievable from the partial ordering semantics;

(ii) the partial ordering semantics must capture all and only the parallelism present in the language, as expressed, for example, through a multiset operational semantics.

In this section, we will discuss the adequacy of our semantics and its relationship with other work with the same objectives, by checking whether they satisfy the above criteria, and by discussing the discriminating power of the proposed behavioural equivalences.

Theorem 4.9 guarantees that our semantics satisfies criterion (i), i.e., there exists a Milner's many step derivation if and only if it corresponds to a linearization of the events of a many step partial ordering derivation. It should be noted that there indeed exists a direct syntactic correspondence between agents and the sets of grapes reachable through derivations, between Milner's rules and ours, and, finally, between the proofs of either derivations. In fact, criterion (i) is shown to hold by a straightforward structural induction. We should like to stress that another by-product of the direct correspondence is that proof techniques developed for the interleaving approach can be borrowed, as done, e.g., in the proofs of Theorems 4.14 and 4.17.

We have not proved criterion (ii), but we claim it. The proof, as shown in [11], would require the introduction of a multiset transition system in which transitions are labelled by multisets of actions, rather than by single actions. The new transition system makes the concurrency of CCS agents explicit by describing the effect of performing concurrent actions simultaneously. The multiset operational semantics can be defined by extending and modifying the inference rule for communication between agents, so that a multiset of actions could be performed and pairs of complementary actions could be synchronized (see [28, 1] for something in this line). Once multiset transitions have been defined, criterion (ii) can be stated as follows. A partial ordering obtained from a computation of an agent contains a set of concurrent events S if and only if there is a multiset derivation the label of which contains all and only the labels of S. The proof of this fact is omitted here since it requires long and tedious work, similar to that used to establish the relationship between the partial ordering and interleaving semantics. More precisely, a mixed ordering which also contains sequences of multisets of actions must be defined; the multiset counterpart of complete concurrency must be proved; and eventually it should be shown that the partial ordering equivalence implies the multiset one.

It is important to note that the criteria (i) and (ii) express the minimal requirements for a partial ordering semantics. Indeed, they only guarantee that the proposed semantics is not in contrast with the interleaving one and that no potential concurrency is lost; nothing is required for ensuring that *all* and *only* the causal relations conveyed by terms are made explicit. A possible way of gaining more confidence in a partial ordering semantics could be checking it against a denotational semantics, based for example on Event Structures [41]. In [12], we have pursued this line by using the distributed operational semantics of [11] to associated Labelled Event Structures to CCS terms. This construction enabled us to prove the consistency of the operational semantics with the denotational one. We have not yet investigated the possiblity of taking a similar approach when starting from the operational semantics of this paper. Certainly, the construction of Labelled Event Structures will be more involved, due to the centralized treatment of choice and recursion.

Once criterion (ii) has been established, it is not difficult to see that multiset equivalence is coarser than our partial ordering equivalence, whichever abstraction mechanism is chosen. Indeed, a multiset equivalence does not respect causal dependencies, as shown by the following example. The agents

$$\alpha.\mathrm{NIL}|\beta.\mathrm{NIL}+\alpha.\beta.\mathrm{NIL} \quad \text{and} \quad \alpha.\mathrm{NIL}|\beta.\mathrm{NIL}$$

are multiset equivalent, but they are not partial ordering equivalent.

Although the example above and Theorem 4.17 may support the choice of partial ordering congruence as the basis for truly concurrent semantics, there are also situations in which this congruence is not completely satisfactory. Indeed, it does not completely respect branching time. For example, the following agents are partial ordering congruent

$$\alpha.(\gamma.\mathrm{NIL}+\delta.\mathrm{NIL})+\alpha.\mathrm{NIL}|\gamma.\mathrm{NIL}+\alpha.\gamma.\mathrm{NIL}$$

$$\alpha.(\gamma.\mathrm{NIL}+\delta.\mathrm{NIL})+\alpha.\mathrm{NIL}|\gamma.\mathrm{NIL}.$$

Nevertheless, the first agent may *cause* via an α either $(\gamma.\mathrm{NIL}+\delta.\mathrm{NIL})$ or just $\gamma.\mathrm{NIL}$, while the second has no choice.

When branching time is felt to be important, alternative approaches can be followed, still using as a starting point the rewriting rules used in this paper.

In [9] we introduced a new partial ordering equivalence which, like the one of this paper, is based on bisimulation but it does respect branching time. The additional discriminating power is gained by resorting to so called Nondeterministic Measurement Systems (NMS), a particular kind of node labelled trees which give an integral representation of the computations of agents. More precisely, an NMS associated to an agent E is a tree the nodes of which are the computations of E, ordered by prefix; its nodes are labelled by the relevant information about the computation performed so far. Two agents are considered as equivalent whenever their NMSs are bisimilar.

Clearly, we can use the labels of the mixed derivations obtained of Definition 4.3 to label NMSs and obtain an observational equivalence which fully respects causality, nondeterminism and their interplay. This equivalence can easily be proved finer than the equivalence of Definition 4.10; and can be proved equivalent to an adaptation to our setting of the equivalence proposed in [43]. For example, the two above agents would be differentiated by the new NMS equivalence, since they generate two NMSs which are not bisimilar. The two NMSs are depicted in Fig. 9, where the nodes are labelled by partial orderings only; both the computations corresponding to the nodes and their generation orderings can be easily inferred.

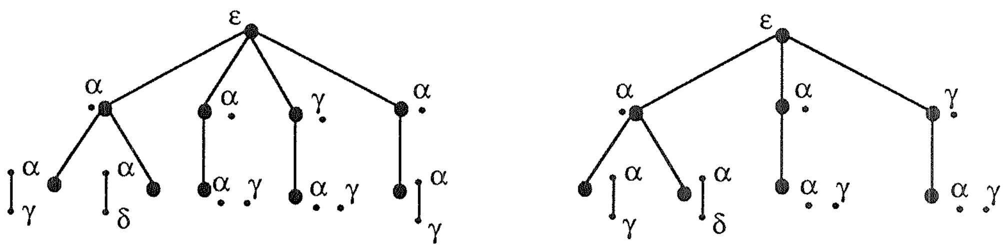


Fig. 9.

There have been earlier attempts to define a partial ordering operational semantics for CCS. However, either proper subsets of CCS have been considered or the interleaving semantics is not the standard CCS one or a formal proof has not been given. Our attempts [8, 9, 10, 11] have already been summarized in the Introduction. We would like to add that in [8], we label every transition with that part of its proof which is needed to recover causality; a similar approach has been followed by in [44] where the whole proofs are used as labels instead. De Cindio et al. [13] map into a subclass of Petri Nets a version of CCS which does not allow the generation of unboundedly many agents in parallel, such as rec x. α.NIL$|\beta.x$. Goltz and Mycroft [19] give a denotational semantics of CCS in terms of Occurrence Nets and an operational semantics in terms of Place/Transitions Nets which does not satisfy criterion (i) (see [10] for an example). Winskel [41, 42] proposes two partial ordering denotational semantics for CCS based on Event Structures and on Petri Nets. He claims that his semantics agrees with Milner's without giving any formal statement of the satisfaction of any criteria similar to (i) and (ii) above. The approach of [10] is refined in [34] to give a distributed account of + and rec; Olderog uses a slightly modified version of our decomposition function and proposes a set of derivation rules very similar to those of [11] to obtain a partial ordering semantics of a language (CCSP) with many similarities to CCS. Satisfaction of criterion (i) is proved, but more involved and less general conditions are stated in place of criterion (ii), and not formally proved.

Recently, Darondeau and Degano [7] proposed a new interleaving-like operational semantics which is nevertheless capable of expressing causality, and provided a complete axiomatization (for finitary CCS) of a congruence based on bisimulation which is as discriminating as the one based on NMSs when labelled by mixed orderings of events.

There are also several papers which aim at providing languages traditionally equipped with interleaving based semantics with partial ordering preserving behavioural equivalences. Castellani and Hennessy [6] provide a fragment of CCS with a semantics based on rewriting rules and bisimulation. Synchronization and restriction are not considered and only single-step derivations are defined. Their observational equivalence does not seem to be comparable with ours even for the common sublanguge. However, the relationships have not yet been fully investigated. In [21], van Glabbeek and Vaandrager propose a Petri Net semantics for finite ACP processes and define two congruence relations (pomset and generalized pomset bisimulation) which seem to coincide with our partial ordering and (partial ordering labelled-) NMS equivalence, respectively. Boudol and Castellani [2] consider an algebra of labelled event structures (without restriction and communication) and define a complete set of axioms for a congruence relation which, we feel, coincides with pomset bisimulation and with the congruence introduced in this paper. None of the above operational approaches considers a language with operators for both recursion and restriction. It is not clear to us how and whether their results could be extended to cope with this significant mixture.

The results presented in this paper certainly require further improvements and extensions. Obviously, the relationships among the various partial ordering equivalences should be assessed, and other notions of equivalence defined and studied. For example, it should be worthwhile to extend to true concurrent models those equivalence or pre-order relations already introduced and proved interesting for interleaving models [15, 33]. More generally, criteria must be established to judge the adequacy and feasibility of equivalence relations for concurrent systems.

Appendix A

The proof of complete concurrency is based on the following steps.

(i) Given a computation, we define its *observation* which extends the notion of mixed derivation of Definition 4.3 in that it is labelled by mixed orderings of events also containing events labelled by τ.

(ii) We show that, given two consecutive *concurrent* events originated by a computation, there always exists another computation which generates them in the reverse order. More precisely, given a two-step computation $\{G_0 I_1 \rightarrow^{[\mu_1, \mathcal{R}_1]} I_1' G_1 I_2 \rightarrow^{[\mu_2, \mathcal{R}_2]} I_2' G_2\}$, the two events originated by it can also be generated in the reverse order, provided that no grape of I'_1 is used by the second partial ordering derivation, namely when $I'_1 \cap I_2 = \emptyset$ (note that $\mathcal{R}_2\downarrow 2 \cap I_2$ may be nonempty).

(iii) We further extend the result above to any set of concurrent events.

(iv) We prove that discarding the events labelled by τ does not affect the overall result.

Recall that two isomorphic ordering of events will be considered identical.

Definition A.1 (*observation*). Given two CCS agents E_0 and E_1, with $\text{dec}(E_0) = G_0$ and $\text{dec}(E_1) = G_n$, and the computation

$$\xi = \{G_0 I_1 \xrightarrow{[\mu_1, \mathcal{R}_1]} I'_1 G_1 \cdots G_{n-1} I_n \xrightarrow{[\mu_n, \mathcal{R}_n]} I'_n G_n\}$$

we call *observation of* ξ the mixed ordering of events $0 = \langle S, l, \leqslant, \lesssim \rangle$, labelled on $\Lambda \cup \{\tau\}$, defined by items (i) and (ii) of Definition 4.3, and by the following

(iii′) $S = S' = \{e_i\}$; $l(e_i) = \mu_i$; restrict F^* to S; and $e_i \lesssim e_j$, $1 \leqslant i \leqslant j \leqslant n$.

Lemma A.2. *Given a two-step computation*

$$\{G_0 I_1 \xrightarrow{[\mu_1, \mathcal{R}_1]} I'_1 G_1 I_2 \xrightarrow{[\mu_2, \mathcal{R}_2]} I'_2 G_2\} \quad \textit{with } I'_1 \cap I_2 = \emptyset,$$

(i) *its observation is* $o = \langle S, l, \leqslant, \lesssim \rangle$, *where*

$$S = \{e_1, e_2\}, \qquad l(e_1) = \mu_1, \quad l(e_2) = \mu_2, \qquad e_1 \leqslant e_1, \quad e_2 \leqslant e_2, \qquad e_1 \lesssim e_2,$$

i.e., the two events are concurrent;

(ii) *there always exists a computation*

$$\{G_0 I_2 \xrightarrow{[\mu_2, \mathcal{R}_1]} I'_2 G_1 I_1 \xrightarrow{[\mu_1, \mathcal{R}_2]} I'_1 G_2\}$$

with observation $o' = \langle S, l, \leqslant, \lesssim' \rangle$, *where* $e_2 \lesssim' e_1$;

(iii) *the same causal dependencies expressed by* F^* *are obtained among the elements of* G_0, *those of* G_2 *and events* e_1 *and* e_2 *while determining the mixed derivations of either* ξ *or* ξ'.

Proof. The proof of the first claim is immediate. Items (ii) and (iii) are proved by induction on the maximal number of steps needed to infer $I_1 \rightarrow^{[\mu_1, \mathcal{R}_1]} I'_1$ and $I_2 \rightarrow^{[\mu_2, \mathcal{R}_2]} I'_2$.

The base case is when $n = 2$. Indeed, in order to have two concurrent events, the deductions of both derivations must make use at least of the axiom (act) and of the inference rule (com) of Definition 3.8 and G_0 can only be of the form $\{\mu_1.E_1 | \text{id}, \text{id} | \mu_2.E_2\}$. The proof of the base case is now immediate.

Let us prove the inductive step by case analysis on the syntatic structure of the grapes in the complete set of grapes G_0 (notice that they have all the same structure). For the sake of brevity, in the rest of the proof, we will call *equivalent* two computations the observations of which differ only in the total ordering, and such that the same causal relation F^* among the elements of G_0, those of G_2, and events e_1 and e_2 holds.

(act): The axiom of Definition 3.8 can never be used in the last step of a proof longer than 2.

(res): If, from

$$\{G_0\ I_1 \xrightarrow{[\mu_1,\mathscr{R}_1]} I_1'\ G_1\ I_2 \xrightarrow{[\mu_2\mathscr{R}_2]} I_2'\ G_2\}$$

we can generate the equivalent computation

$$\{G_0\ I_2 \xrightarrow{[\mu_2,\underline{\mathscr{R}}_1]} I_2'\ \underline{G}_1\ I_1 \xrightarrow{[\mu_1,\underline{\mathscr{R}}_2]} I_1'\ G_2\},$$

then from

$$\{G_0\backslash\alpha\ \ I_1\backslash\alpha \xrightarrow{[\mu_1,\mathscr{R}_1\backslash\alpha]} I_1'\backslash\alpha\ \ G_1\backslash\alpha\ \ I_1\backslash\alpha \xrightarrow{[\mu_2,\mathscr{R}_2\backslash\alpha]} I_2'\backslash\alpha\ \ G_2\backslash\alpha\}$$

we generate the equivalent computation

$$\{G_0\backslash\alpha\ \ I_2\backslash\alpha \xrightarrow{[\mu_2,\underline{\mathscr{R}}_1\backslash\alpha]} I_2'\backslash\alpha\ \ \underline{G}_1\backslash\alpha\ \ I_1\backslash\alpha \xrightarrow{[\mu_1,\underline{\mathscr{R}}_2\backslash\alpha]} I_1'\backslash\alpha\ \ G_2\backslash\alpha\}$$

where the length of the deductions of both $I_1 \rightarrow^{[\mu_1,\underline{\mathscr{R}}_2]} I_1'$ and $I_2 \rightarrow^{[\mu_2,\underline{\mathscr{R}}_2]} I_2'$ is increased by 1.

Analogously for (rel).

(sum): If, from

$$\{\mathrm{dec}(E_1)I_1 \xrightarrow{[\mu_1,\mathscr{R}_1]} I_1'\ G_1\ I_2 \xrightarrow{[\mu_2,\mathscr{R}_2]} I_2'\ G_2\},$$

with $I_1 = \mathrm{dec}(E_1 - I_3)$, we can generate the equivalent computation

$$\{\mathrm{dec}(E_1 - \underline{I}_3)I_2 \xrightarrow{[\mu_2,\underline{\mathscr{R}}_1]} I_2'\ \underline{G}_1\ I_1 \xrightarrow{[\mu_1,\underline{\mathscr{R}}_2]} I_1'\ G_2\},$$

with $I_1 = \mathrm{dec}(E_1 - \underline{I}_3)$, then from

$$\{\{E_1+E\}\{E_1+E\} \xrightarrow{[\mu_1,\{E_1+E\}\leqslant(I_3\cup\mathscr{R}_1\downarrow 2)]} I_2'\ G_1\ I_2 \xrightarrow{[\mu_2,\mathscr{R}_2]} I_2'\ G_2\}$$

we generate the equivalent computation

$$\{\{E_1+E\}\{E_1+E\} \xrightarrow{[\mu_2,\{E_1+E\}\leqslant(\underline{I}_3\cup\underline{\mathscr{R}}_1\downarrow 2)]} I_2'\ \underline{G}_1\ I_1 \xrightarrow{[\mu_1,\underline{\mathscr{R}}_2]} I_1'\ G_2\}.$$

(com): The inductive step for the two first (com) rules can be proved following the same pattern of the proof of case (res). Let us consider the most complicated case of the third implication: both events observed from the given computation are labelled by τ. Two further cases may arise.

(i) In the first case we inductively assume that (indexes l and r stand for *left* and *right*) from

$$\{G_0^{\mathrm{l}}\ I_1^{\mathrm{l}} \xrightarrow{[\lambda_1,\mathscr{R}_1^{\mathrm{l}}]} I_1'^{\mathrm{l}}\ G_1^{\mathrm{l}}\ I_2^{\mathrm{l}} \xrightarrow{[\lambda_2,\mathscr{R}_2^{\mathrm{l}}]} I_2'^{\mathrm{l}}\ G_2^{\mathrm{l}}\}$$

we can generate the equivalent computation

$$\{G_0^{\mathrm{l}}\ I_2^{\mathrm{l}} \xrightarrow{[\lambda_2,\mathcal{R}_1^{\mathrm{l}}]} I_2'^{\mathrm{l}}\ \underline{G}_1^{\mathrm{l}}\ I_1^{\mathrm{l}} \xrightarrow{[\lambda_1,\mathcal{R}_2^{\mathrm{l}}]} I_1'^{\mathrm{l}}\ G_2^{\mathrm{l}}\}$$

and that from

$$\{G_0^{\mathrm{r}}\ I_1^{\mathrm{r}} \xrightarrow{[\lambda_1^-,\mathcal{R}_1^{\mathrm{r}}]} I_1'^{\mathrm{r}}\ G_1^{\mathrm{r}}\ I_2^{\mathrm{r}} \xrightarrow{[\lambda_2^-,\mathcal{R}_2^{\mathrm{r}}]} I_2'^{\mathrm{r}}\ G_2^{\mathrm{r}}\}$$

we can generate the equivalent computation

$$\{G_0^{\mathrm{r}}\ I_2^{\mathrm{r}} \xrightarrow{[\lambda_2^-,\mathcal{R}_1^{\mathrm{r}}]} I_2'^{\mathrm{r}}\ \underline{G}_1^{\mathrm{r}}\ I_1^{\mathrm{r}} \xrightarrow{[\lambda_1^-,\mathcal{R}_2^{\mathrm{r}}]} I_1'^{\mathrm{r}}\ G_2^{\mathrm{r}}\}.$$

By applying the third case of rule (com), we obtain from

$$\{G_0^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|G_0^{\mathrm{r}}\ \ I_1^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|I_1^{\mathrm{r}} \xrightarrow{[\tau,\mathcal{R}_1^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|\mathcal{R}_1^{\mathrm{r}}]} I_1'^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|I_1'^{\mathrm{r}}$$

$$G_1^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|G_1^{\mathrm{r}}\ \ I_2^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|I_2^{\mathrm{r}} \xrightarrow{[\tau,\mathcal{R}_2^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|\mathcal{R}_2^{\mathrm{r}}]} I_2'^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|I_2'^{\mathrm{r}}\ \ G_2^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|G_2^{\mathrm{r}}\},$$

we generate the equivalent computation

$$\{G_0^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|G_0^{\mathrm{r}}\ \ I_2^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|I_2^{\mathrm{r}} \xrightarrow{[\tau,\mathcal{R}_1^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|\mathcal{R}_1^{\mathrm{r}}]} I_2'^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|I_2'^{\mathrm{r}}$$

$$\underline{G}_1^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|\underline{G}_1^{\mathrm{r}}\ \ I_1^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|I_1^{\mathrm{r}} \xrightarrow{[\tau,\mathcal{R}_2^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|\mathcal{R}_2^{\mathrm{r}}]} I_1'^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|I_1'^{\mathrm{r}}\ \ G_2^{\mathrm{l}}|\mathrm{id}\cup\mathrm{id}|G_2^{\mathrm{r}}\}.$$

(ii) The proof is straightforward in the other case which occurs when we inductively assume that from

$$\{G_0^{\mathrm{l}}\ I_1^{\mathrm{l}} \xrightarrow{[\lambda_1,\mathcal{R}_1^{\mathrm{l}}]} I_1'^{\mathrm{l}}\ G_1^{\mathrm{l}}\ I_2^{\mathrm{l}} \xrightarrow{[\lambda_2,\mathcal{R}_2^{\mathrm{l}}]} I_2'^{\mathrm{l}}\ G_2^{\mathrm{l}}\}$$

we can generate the equivalent computation

$$\{G_0^{\mathrm{l}}\ I_2^{\mathrm{l}} \xrightarrow{[\lambda_2,\mathcal{R}_1^{\mathrm{l}}]} I_2'^{\mathrm{l}}\ \underline{G}_1^{\mathrm{l}}\ I_1^{\mathrm{l}} \xrightarrow{[\lambda_1,\mathcal{R}_2^{\mathrm{l}}]} I_1'^{\mathrm{l}}\ G_2^{\mathrm{l}}\}$$

and that

$$\{G_0^{\mathrm{r}}\ I_1^{\mathrm{r}} \xrightarrow{[\lambda_2^-,\mathcal{R}_1^{\mathrm{r}}]} I_1'^{\mathrm{r}}\ G_1^{\mathrm{r}}\ I_2^{\mathrm{r}} \xrightarrow{[\lambda_1^-,\mathcal{R}_2^{\mathrm{r}}]} I_2'^{\mathrm{r}}\ G_2^{\mathrm{r}}\}.$$

(rec): Analogous to the proof of (sum).

Corollary A.3 (two consecutive concurrent events can be generated in either ordering). *Given a computation ξ with observation $o = \langle S, l, \leqslant, \lesssim \rangle$, let e and e' be two concurrent events generated according to Definition* 4.3 *in correspondence to two consecutive occurrences of partial ordering derivations. There always exist a computation*

ξ' with observation $o' = \langle S, l, \leqslant, \lesssim' \rangle$, where

$$\lesssim' = (\lesssim - \{e \lesssim e'\}) \cup \{e' \lesssim' e\}.$$

Proof. Let

$$\xi = \{G_0 \cdots G_{i-1}\; I_i \xrightarrow{[\mu_i, \mathscr{R}_i]} I'_i\; G_i\; I_{i+1} \xrightarrow{[\mu_{i+1}, \mathscr{R}_{i+1}]} I'_{i+1}\; G_{i+1} \cdots G_n\}$$

where events e and e' are originated by the ith and $(i+1)$th partial ordering derivations. By applying Lemma A.2 it is easy to construct the required computation which is as follows.

$$\xi' = \{G_0 \cdots G_{i-1}\; \underline{I}_i \xrightarrow{[\mu_{i+1}, \underline{\mathscr{R}}_i]} \underline{I}'_i\; \underline{G}_i\; \underline{I}_{i+1} \xrightarrow{[\mu_i, \underline{\mathscr{R}}_{i+1}]} \underline{I}'_{i+1}\; G_{i+1} \cdots G_n\}$$

The only check to be performed is that the partial ordering of the observation of ξ' is indeed $\leqslant$. This follows immediately by noticing that the causal dependences F^* obtained among the elements of G_{i-1}, those of G_{i+1} and the events while determining the mixed derivation of

$$\{G_{i-1}\; I_i \xrightarrow{[\mu_i, \mathscr{R}_i]} I'_i\; G_i\; I_{i+1} \xrightarrow{[\mu_{i+1}, \mathscr{R}_{i+1}]} I'_{i+1}\; G_{i+1}\}$$

are, by Lemma A.2(iii), exactly the same causal dependencies set while determining the mixed derivations of

$$\{G_{i-1}\; \underline{I}_i \xrightarrow{[\mu_{i+1}, \underline{\mathscr{R}}_i]} \underline{I}'_i\; \underline{G}_i\; \underline{I}_{i+1} \xrightarrow{[\mu_i, \underline{\mathscr{R}}_{i+1}]} \underline{I}'_{i+1}\; G_{i+1}\}. \qquad \square$$

Lemma A.4 (two concurrent events can be generated in either ordering). *Given two CCS agents E_0 and E_1 and an observation $o = \langle S, l, \leqslant, \lesssim \rangle$ such that there exists a computation from E_0 to E_1 with observation o, we have that for all $\lesssim'$ such that $\leqslant \subseteq \lesssim'$, there exists a computation from E_0 to E_1, with observation $o' = \langle S, l, \leqslant, \lesssim' \rangle$.*

Proof. Let Ξ be the set of all the computations from E_0 to E_1 originating the same partial ordering of events $\langle S, l, \leqslant \rangle$ according to Definition 4.3. We have to prove that, given the partial ordering of events $\langle S, l, \leqslant \rangle$ and a total ordering $\lesssim' = \{e_i \lesssim' e_j \mid i \leqslant j\} = e_0\, e_1 \cdots e_k$ on the events of S such that $\leqslant \subseteq \lesssim'$, there exists a computation $\xi' \in \Xi$ with observation $o' = \langle S, l, \leqslant, \lesssim' \rangle$. Let ξ^0 be a computation in Ξ. We construct a sequence of computations $\{\xi^0, \xi^1, \ldots, \xi'\}$, all in Ξ, with the intuition that ξ^h originates a total ordering the first h elements of which are the same h first elements of $\lesssim'$ (of ξ').

Assume that $\xi^{j_n} \in \Xi$ has observation $o^{j_n} = \langle S, l, \leqslant, \lesssim^{j_n} \rangle$. If $\lesssim^{j_n} = \lesssim'$, the required computation is found. Otherwise, assume inductively that $\lesssim^{j_n}$ has the same n first elements $\lesssim'$ has, and that e_n occurs as the $(m+1)$th element, i.e., $\lesssim^{j_n} = e_0\, e_1 \cdots e_{n-1}\, e'_n \cdots e'_{m-1}\, e_n \cdots e_k$. Using Corollary A.3 it is easy to construct a computation ξ^{j_n+1} with observation $\langle S, l, \leqslant, \lesssim^{j_n+1} \rangle$ where $\lesssim^{j_n+1} = e_0\, e_1 \ldots e_{n-1}\, e'_n \ldots e_n\, e'_{m-1} \ldots e_k$. In fact e_n and e'_{m-1} are concurrent, for they appear in reverse order in $\lesssim^{j_n}$ and $\lesssim'$, which both contain $\leq$. Performing a total of $m-n$ exchanges we obtain an observation the total ordering of which is $\lesssim^{j_n+m-n} = \lesssim^{j_n+1}$, and the inductive step is proved. □

Theorem 4.6 (complete concurrency). *Given two CCS agents E_0 and E_1 and a mixed ordering of events $d = \langle S, l, \leq, \lesssim\rangle$ such that $E_0 \Rrightarrow^d E_1$, we have that*

- $\leq \subseteq \lesssim$;
- $\forall \lesssim'$ *such that* $\leq \subseteq \lesssim'$, *there exist a mixed derivation* $E_0 \Rrightarrow^{d'} E_1$, *with* $d' = \langle S, l, \leq, \lesssim'\rangle$.

Proof. Let ξ be a computation, $o = \langle S^o, l^o, \leq^o, \lesssim^o\rangle$ be its observation and $d = \langle S, l, \leq, \lesssim\rangle$ be its mixed ordering of events. We have that $S = S^o - \{e \mid l(e) = \tau\}$, since Definition A.1 is obtained from Definition 4.3 by modifying item (iii) only, used there to discard events labelled by τ and to accordingly restrict l, $\leq$, $\lesssim$. The first claim is obvious. We are left to prove the second claim: a total ordering $\lesssim'$ on the events of S is given such that $\leq \subseteq \lesssim'$, and we must find a computation ξ' with mixed ordering of events $d' = \langle S, l, \leq, \lesssim'\rangle$. It suffices to find a total ordering $\lesssim^{o'}$ such that $\leq^0 \subseteq \lesssim^{0'}$, and its restriction to S is $\lesssim'$; this is because Lemma A.4 can then be applied. Such a $\lesssim^{o'}$ does exist, since relation $R = \lesssim' \cup \leq^o$ is a partial ordering (only the events labelled by τ may be unrelated). In fact, a cycle in R would imply the existence of a cycle either in $\lesssim'$ or in $\leq^o$, for $\leq \subseteq \lesssim'$. Indeed, we can choose as $\lesssim^{o'}$ any totalization of R, obtained by adding the necessary pairs $\mu \lesssim^{o'} \tau$ or $\tau \lesssim^{o'} \mu$, and removing reflexivity. □

Appendix B

Theorem 4.14. *Partial ordering observational equivalence is preserved by operators* NIL, $\mu.$, $\backslash\alpha$, $[\phi]$ *and* $|$.

Proof. Let E, E_0, E_1 be CCS agents. The proof proceeds by case analysis on the operators of CCS, under the hypothesis that there exists a bisimulation R containing the pair $\langle E_0, E_1\rangle$, i.e., $E_0 \cong E_1$.

(act): It suffices to prove that adding to R the pair the pair $\langle \mu E_0, \mu E_1\rangle$ results in a bisimulation. We distinguish two cases. If $\mu \neq \tau$, let us consider a computation for which $\mu E_0 \Rrightarrow^{\langle S, L, \leq\rangle} E_0'$, and call e the event corresponding to its first quadruple $\{\mu E_0\} \rightarrow^{[\mu, \emptyset]} \mathrm{dec}(E_0)$. There exists then another computation such that $E_0 \Rrightarrow^{\langle S', l, \leq'\rangle} E_0'$, with $S' = S - \{e\}$ and $\leq' = \leq - (\{e \leq e\} \cup \{e \leq e' \mid e' \in S'\})$. By hypothesis, we can always grow a computation for which $E_1 \Rrightarrow^{\langle S', l, \leq'\rangle} E_1'$, with $E_0' \cong E_1'$; and from this the required computation such that $\mu E_1 \Rrightarrow^{\langle S, l, \leq\rangle} E_1'$. And vice versa. If $\mu = \tau$, obvious.

(res): We have to prove that $E_0\backslash\alpha \cong E_1\backslash\alpha$. The proof is easy: if there exists an agent E_0' (of the form $E\backslash\alpha$) such that $E_0\backslash\alpha \Rrightarrow^h E_0'$ (i.e., for whichever computation you choose from $\text{dec}(E_0)$ to $\text{dec}(E_0')$ with no quadruples of the form $I \to^{[\alpha,\mathcal{R}]} I'$), by hypothesis $E_0 \cong E_1$ we can always find an agent E_1' such that $E_0' \cong E_1'$ and $E_1\backslash\alpha \Rightarrow^h E_1'$ (i.e., there exists a computation from $\text{dec}(E_1)$ to $\text{dec}(E_1')$ with no quadruples of the form $I \to^{[\alpha,\mathcal{R}]} I'$). And vice versa. Thus, the required bisimulation is $R' = \{\langle E\backslash\alpha, E'\backslash\alpha\rangle \mid \langle E, E'\rangle \in R\}$.

(rel): Trivial, since ϕ is a permutation of $\Lambda \cup \{\tau\}$ that preserves τ and $^-$: the required bisimulation is $R' = \{\langle E_0[\phi], E_1[\phi]\rangle \mid \langle E_0, E_1\rangle \in R\}$.

(com): We only consider the case of right $|$ context; the other case is symmetrical. The required bisimulation is $R' = \{\langle E_0 | E, E_1 | E\rangle \mid \langle E_0, E_1\rangle \in R\}$. In order to support our claim, we now prove that whenever $E_0 | E \Rightarrow^h E_0' | E'$ then $E_1 | E \Rightarrow^h E_1' | E'$ and $E_0' | E' \cong E_1' | E'$. By a symmetric argument, R' is therefore a bisimulation. This is the most difficult case to be proved, and, in order to guide the reader in understanding the proof, we first consider the case when there is no communication between E_0 and E. Then, we prove the thesis for a single-step computation consisting of a synchronization. Finally, we extend this result to the general case.

In the first case, for every computation of $E_0 | E$ with no communication

$$\xi_0 = \{\text{dec}(E_0) | \text{id} \cup \text{id} | \text{dec}(E) \;\; I_1 \xrightarrow{[\mu_1, \mathcal{R}_1]} I_1' \; G_1 \cdots$$

$$G_{n-1} \; I_n \xrightarrow{[\mu_n, \mathcal{R}_n]} I_n' \;\; \text{dec}(E_0') | \text{id} \cup \text{id} | \text{dec}(E')\}$$

with $h = \langle S, l, \leqslant\rangle$ as label of its partial ordering many step derivation, we must find a computation of $E_1 | E$ with no communication

$$\xi_1 = \{\text{dec}(E_1) | \text{id} \cup \text{id} | \text{dec}(E)\} \;\; \underline{I}_1 \xrightarrow{[\underline{\mu}_1, \underline{\mathcal{R}}_1]} \underline{I}_1' \; \underline{G}_1 \cdots$$

$$\underline{G}_{m-1} \; \underline{I}_m \xrightarrow{[\underline{\mu}_m, \underline{\mathcal{R}}_m]} \underline{I}_m' \;\; \text{dec}(E_1') | \text{id} \cup \text{id} | \text{dec}(E')\},$$

with the same h as label of the partial ordering many step derivation, and $E_0' | E' \cong E_1' | E'$.

We can write each (occurrence of) complete set of grapes G_k as $G_k^{\text{l}} | \text{id} \cup \text{id} | G_k^{\text{r}}$ (index l is for *left*, r for *right*), where $G_k^{\text{l}}(G_k^{\text{r}})$ is a(n occurrence of) complete set of grapes to which $\text{dec}(E_0)(\text{dec}(E))$ has evolved. Now, since there is no communication, it is possible to partition ξ_0 in two parts: the first one contains those quadruples involving only grapes in $G_k^{\text{l}} | \text{id}$; the second part contains those quadruples involving only grapes in $\text{id} | G_k^{\text{r}}$. This can always be done, by looking whether $I_k \subseteq G_k^{\text{l}} | \text{id}$, or $I_k \subseteq \text{id} | G_k^{\text{r}}$.

The partition above induces a partition on h in h^{l} and h^{r}, as well. The events of h are then accordingly partitioned depending on whether they correspond to the quadruples in the left or the right part of ξ_0, respectively. It is important to note that, since $I_k \cap I_j = \emptyset$, for all $I_k \subseteq G_k^{\text{l}} | \text{id}$, $I_j \subseteq \text{id} | G_k^{\text{r}}$, all events of h^{l} are *concurrent* with those of h^{r}.

Two computations can now be generated from ξ_0, by "splitting" each quadruple in its premises, more precisely

- $I_j \rightarrow^{[\mu_j,\mathcal{R}_j]} I'_j$, with $I_j \subseteq G^l_j|\mathrm{id}$, originates $I^l_j \rightarrow^{[\mu_j,\mathcal{R}^l_j]} I'^l_j$, where $\mathcal{R}^l_j|\mathrm{id}=\mathcal{R}_j$, $I^l_j|id=I_j$, $I'^l_j|\mathrm{id}=I'_j$;
- $I_j \rightarrow^{[\mu_j,\mathcal{R}_j]} I'_j$, with $I_j \subseteq \mathrm{id}|G^r_j$, originates $I^r_j \rightarrow^{[\mu_j,\mathcal{R}^r_j]} I'^r_j$, where $\mathrm{id}|\mathcal{R}^r_j=\mathcal{R}_j$, $\mathrm{id}|I^r_j=I_j$, $\mathrm{id}|I'^l_j=I'_j$.

We obtain the following computations

$$\xi^l_0=\{\mathrm{dec}(E_0)\ I^l_1 \xrightarrow{[\mu^l_1,\mathcal{R}^l_1]} I'^l_1\ G^l_1 \cdots G^l_{p-1}\ I^l_p \xrightarrow{[\mu^l_p,\mathcal{R}^l_p]} I'^l_p\ \mathrm{dec}(E'_0)\}$$

$$\xi^r_0=\{\mathrm{dec}(E)\ I^r_1 \xrightarrow{[\mu^r_1,\mathcal{R}^r_1]} I'^r_1\ G^r_1 \cdots G^r_{q-1}\ I^r_q \xrightarrow{[\mu^r_q,\mathcal{R}^r_q]} I'^r_q\ \mathrm{dec}(E')\}$$

which give rise to partial ordering derivations which are (isomorphic to) h^l and h^r: $h^x=\langle S^x, l^x, \leqslant^x\rangle$ (with $x=\mathrm{l}, \mathrm{r}$), where

- $S^x=\{e_j\in S\,|\,e_j$ is generated accordingly to Definition 4.3 in correspondence to $I^x_j \rightarrow^{[\mu_j,\mathcal{R}^x_j]} I'^x_j\}$;
- l^x is the restriction of 1 to S^x;
- $\leqslant^x=\{e_j\leqslant e_k\,|\,e_j, e_k\in S^x\}$.

By hypothesis, E_0 and E_1 are partial ordering equivalent, thus we can find an agent E'_1 partial ordering equivalent to E'_0, such that $E_1 \Rightarrow^{h^l} E'_1$. This partial ordering derivation is obtained by a computation, say

$$\xi^l_1=\{\mathrm{dec}(E_1)\ \underline{I}^l_1 \xrightarrow{[\underline{\mu}^l_1,\underline{\mathcal{R}}^l_1]} \underline{I}'^l_1\ \underline{G}^l_1 \cdots \underline{G}^l_{z-1}\ \underline{I}^l_z \xrightarrow{[\underline{\mu}_z,\underline{\mathcal{R}}^l_z]} \underline{I}'^l_z\ \mathrm{dec}(E'_1)\}.$$

Eventually, we can "put Humpty together again" first by inferring from quadruple

- $\underline{I}^l_j \rightarrow^{[\underline{\mu}^l_j,\underline{\mathcal{R}}^l_j]} \underline{I}'^l_j$, $1\leqslant j\leqslant z$, the quadruple

$$\underline{I}^l_j|\mathrm{id} \xrightarrow{[\underline{\mu}^l_j,\underline{\mathcal{R}}^l_j|\mathrm{id}]} \underline{I}'^l_j|\mathrm{id}=\underline{I}_j \xrightarrow{[\underline{\mu}^l_j,\underline{R}_j]} \underline{I}'_j,$$

- $I^r_j \rightarrow^{[\mu^r_j,\mathcal{R}^r_j]} I'^r_j$, $1\leqslant j\leqslant q$, the quadruple

$$\mathrm{id}|I^r_j \xrightarrow{[\mu^r_j,\mathrm{id}|\mathcal{R}_j]} \mathrm{id}|I'^r_j=I_j \xrightarrow{[\mu^r_j,\mathcal{R}_j]} I'_j;$$

and then by generating the following computation.

$$\xi_1=\{\mathrm{dec}(E_1)|\mathrm{id}\cup\mathrm{id}|\mathrm{dec}(E)\ \underline{I}_1 \xrightarrow{[\underline{\mu}^l_1,\underline{\mathcal{R}}_1]} \underline{I}'_1\ \underline{G}^l_1\cup\mathrm{id}|\mathrm{dec}(E)\cdots$$

$$\cdots\underline{I}_z \xrightarrow{[\underline{\mu}_z,\underline{\mathcal{R}}_z]} \underline{I}'_z\ \mathrm{dec}(E'_1)|\mathrm{id}\cup\mathrm{id}|\mathrm{dec}(E)\quad(\text{coming from }\xi^l_1),$$

$$I_1 \xrightarrow{[\mu^r_1,\mathcal{R}_1]} I'_1\ \underline{G}^l_z|\mathrm{id}\cap\mathrm{id}|G^r_1\cdots$$

$$\cdots I_q \xrightarrow{[\mu^r_q,\mathcal{R}_q]} I'_q\ \mathrm{dec}(E'_1)|\mathrm{id}\cup\mathrm{id}|\mathrm{dec}(E')\}\quad(\text{coming from }\xi^r_0).$$

Computation ξ_1 is indeed such that $E_1|E \Rightarrow^h E'_1|E'$, and thus the proof is completed in the case of no communication. Actually, Theorem 4.6 stating complete concurrency makes it possible to compose computations ξ_1^{l} and ξ_0^{r} because the quadruples in the former generate events which are *concurrent* with all those generated by the quadruples in the latter.

We now consider a computation containing only a single quadruple resulting from a synchronization, i.e.,

$$\xi_0 = \{\mathrm{dec}(E_0)\,|\,\mathrm{id} \cup \mathrm{id}\,|\,\mathrm{dec}(E)\ \ I \xrightarrow{[\tau,\mathcal{R}]} I'\ \ \mathrm{dec}(E_0')\,|\,\mathrm{id} \cup \mathrm{id}\,|\,\mathrm{dec}(E')\}$$

with empty partial ordering as label of its derivation. We must find a computation of $E_1\,|\,E$

$$\xi_1 = \{\mathrm{dec}(E_1)\,|\,\mathrm{id} \cup \mathrm{id}\,|\,\mathrm{dec}(E)\ \ \underline{I} \xrightarrow{[\tau,\mathcal{R}]} \underline{I}'\ \ \mathrm{dec}(E_1')\,|\,\mathrm{id} \cup \mathrm{id}\,|\,\mathrm{dec}(E')\},$$

with the same empty partial ordering labelling its derivation, $E_0'\,|\,E' \cong E_1'\,|\,E'$. Again, two computations can be generated, by "splitting" the quadruple in its premises, more precisely $I \to^{[\tau,\mathcal{R}]} I'$ originates $I^{\mathrm{l}} \to^{[\lambda,\mathcal{R}^{\mathrm{l}}]} I'^{\mathrm{l}}$ and $I^{\mathrm{r}} \to^{[\lambda^-,\mathcal{R}^{\mathrm{r}}]} I'^{\mathrm{r}}$, where $I^{\mathrm{l}}\,|\,\mathrm{id} \cup \mathrm{id}\,|\,I^{\mathrm{r}} = I$, $I'^{\mathrm{l}}\,|\,\mathrm{id} \cup \mathrm{id}\,|\,I'^{\mathrm{r}} = I'$, and $\mathcal{R}^{\mathrm{l}}\,|\,\mathrm{id} \cup \mathrm{id}\,|\,\mathcal{R}^{\mathrm{r}} = \mathcal{R}$.

We obtain the following computations

$$\xi_0^{\mathrm{l}} = \{\mathrm{dec}(E_0)\ I^{\mathrm{l}} \xrightarrow{[\lambda,\mathcal{R}^{\mathrm{l}}]} I'^{\mathrm{l}}\,\mathrm{dec}(E_0')\};$$

$$\xi_0^{\mathrm{r}} = \{\mathrm{dec}(E)\ I^{\mathrm{r}} \xrightarrow{[\lambda^-,\mathcal{R}^{\mathrm{r}}]} I'^{\mathrm{r}}\,\mathrm{dec}(E')\},$$

that generate one new event each, say, e^{l} and e^{r}, and have the following partial ordering derivations

$$h_\tau^{\mathrm{l}} = \langle \{e^{\mathrm{l}}\}, \{l(e^{\mathrm{l}}) = \lambda), \{e^{\mathrm{l}} \leq e^{\mathrm{l}}\}\rangle$$

$$h_\tau^{\mathrm{r}} = \langle \{e^{\mathrm{r}}\}, \{l(e^{\mathrm{r}}) = \lambda^-\}, \{e^{\mathrm{r}} \leq e^{\mathrm{r}}\}\rangle.$$

By inductive hypothesis E_0 and E_1 are congruent, thus we can find an agent E_1' equivalent to E_0', such that $E_1 \Rightarrow^{h_\tau^{\mathrm{l}}} E_1'$. This partial ordering derivation is obtained by a computation, say

$$\xi_1^{\mathrm{l}} = \{\mathrm{dec}(E_1)\ \underline{I}^{\mathrm{l}} \xrightarrow{[\lambda,\mathcal{R}^{\mathrm{l}}]} \underline{I}'^{\mathrm{l}}\,\mathrm{dec}(E_1')\}.$$

We can now "put Humpty together again" first by inferring from quadruples

$$\underline{I}^{\mathrm{l}}\,|\,\mathrm{id} \xrightarrow{[\lambda,\mathcal{R}^{\mathrm{l}}|\mathrm{id}]} \underline{I}'^{\mathrm{l}}\,|\,\mathrm{id} \ \text{ and } \ \mathrm{id}\,|\,I^{\mathrm{r}} \xrightarrow{[\lambda^-,\mathrm{id}|\mathcal{R}^{\mathrm{r}}]} \mathrm{id}\,|\,I'^{\mathrm{r}}$$

the quadruple

$$\underline{I}^{\mathrm{l}}\,|\,I^{\mathrm{r}} \xrightarrow{[\tau,\mathcal{R}^{\mathrm{l}},\mathcal{R}^{\mathrm{r}}]} \underline{I}'^{\mathrm{l}}\,|\,\underline{I}'^{\mathrm{r}} = \underline{I} \xrightarrow{[\tau,\mathcal{R}]} \underline{I}'$$

and then by generating the following computation

$$\xi_1 = \{\mathrm{dec}(E_1)\,|\,\mathrm{id} \cup \mathrm{id}\,|\,\mathrm{dec}(E)\ \underline{I} \xrightarrow{[\tau,\underline{\mathscr{R}}]} \underline{I}'\ \mathrm{dec}(E_1')\,|\,\mathrm{id} \cup \mathrm{id}\,|\,\mathrm{dec}(E')\}.$$

Obviously $E_1 \,|\, E \Rrightarrow^h E_1' \,|\, E'$.

Now we can better face the general case, by using the facts proved above. Suppose we are given the following computation

$$\xi_0 = \mathscr{A}_0 I_1 \xrightarrow{[\tau_1,\mathscr{R}_1]} I_1' \mathscr{A}_1 I_2 \xrightarrow{[\tau_2,\mathscr{R}_2]} I_2' \cdots I_n \xrightarrow{[\tau_n,\mathscr{R}_n]} I_n' \mathscr{A}_n$$

where $\mathscr{A}_0, \mathscr{A}_1, \ldots, \mathscr{A}_n$ denote (possibly empty) segments of computation without communications, and the invisible actions due to the n communications carry indexes in order to uniquely pick them up. We can split ξ_0, as we did in the two cases above, obtaining the following computations.

$$\xi_0^{\mathrm{l}} = \mathscr{A}_0^{\mathrm{l}} I_1^{\mathrm{l}} \xrightarrow{[\lambda_1,\mathscr{R}_1^{\mathrm{l}}]} I_1'^{\mathrm{l}} \mathscr{A}_1^{\mathrm{l}} I_2^{\mathrm{l}} \xrightarrow{[\lambda_2,\mathscr{R}_2^{\mathrm{l}}]} I_2'^{\mathrm{l}} \cdots I_n^{\mathrm{l}} \xrightarrow{[\lambda_n,\mathscr{R}_n^{\mathrm{l}}]} I_n'^{\mathrm{l}} \mathscr{A}_n^{\mathrm{l}}$$

$$\xi_0^{\mathrm{r}} = \mathscr{A}_0^{\mathrm{r}} I_1^{\mathrm{r}} \xrightarrow{[\lambda_1^-,\mathscr{R}_1^{\mathrm{r}}]} I_1'^{\mathrm{r}} \mathscr{A}_1^{\mathrm{r}} I_2^{\mathrm{r}} \xrightarrow{[\lambda_2^-,\mathscr{R}_2^{\mathrm{r}}]} I_2'^{\mathrm{r}} \cdots I_n^{\mathrm{r}} \xrightarrow{[\lambda_n^-,\mathscr{R}_n^{\mathrm{r}}]} I_n'^{\mathrm{r}} \mathscr{A}_n^{\mathrm{r}}.$$

Computations ξ_0^{l} and ξ_0^{r} originate partial ordering derivations labelled by the partial orderings h^{l} and h^{r}, respectively. As done before, we can obtain the following computation originating, by inductive hypothesis, a partial ordering derivation with partial ordering h^{l} as label.

$$\xi_1^{\mathrm{l}} = \underline{\mathscr{A}}_0^{\mathrm{l}}\, \underline{I}_1 \xrightarrow{[\underline{\lambda}_1,\underline{\mathscr{R}}_1^{\mathrm{l}}]} \underline{I}_1'^{\mathrm{l}}\, \underline{\mathscr{A}}_1^{\mathrm{l}}\, \underline{I}_2^{\mathrm{l}} \xrightarrow{[\underline{\lambda}_2,\underline{\mathscr{R}}_2^{\mathrm{l}}]} \underline{I}_2'^{\mathrm{l}} \cdots$$

$$\cdots \underline{I}_n^{\mathrm{l}} \xrightarrow{[\underline{\lambda}_n,\underline{\mathscr{R}}_n^{\mathrm{l}}]} \underline{I}_n'^{\mathrm{l}}\, \underline{\mathscr{A}}_n^{\mathrm{l}}.$$

We may now compose ξ_1^{l} and ξ_0^{r}, to obtain ξ_1, by iteratively interleaving their parts without communication and "synchronizing" the quadruples with action λ_i and $\underline{\lambda}_i$. in doing so, two cases may arise, depending on whether the actions used for synchronization are generated in the same order or not. More accurately, whether $\underline{\lambda}_i = (\lambda_i^-)^-$ or $\underline{\lambda}_i \neq (\lambda_i^-)^-$. In the first case, no trouble arises and the required computation is simply

$$\xi_1 = \mathscr{A}_0^{\mathrm{l}} \mathscr{A}_0^{\mathrm{r}}\, \underline{I}_1^{\mathrm{l}} \,|\, I_1^{\mathrm{r}} \xrightarrow{[\tau_1,\underline{\mathscr{R}}_1^{\mathrm{l}}|\mathscr{R}_1^{\mathrm{r}}]} \underline{I}_1'^{\mathrm{l}} \,|\, I_1'^{\mathrm{r}}\, \mathscr{A}_1^{\mathrm{l}} \mathscr{A}_1^{\mathrm{r}} \cdots$$

$$\cdots \underline{I}_n^{\mathrm{l}} \,|\, I_n^{\mathrm{r}} \xrightarrow{[\tau_n,\underline{\mathscr{R}}_n^{\mathrm{l}}|\mathscr{R}_1^{\mathrm{r}}]} \underline{I}_n'^{\mathrm{l}} \,|\, I_n'^{\mathrm{r}}\, \mathscr{A}_n^{\mathrm{l}} \mathscr{A}_n^{\mathrm{r}}$$

where $\underline{I}_i^{\mathrm{l}} \,|\, I_i^{\mathrm{r}} \rightarrow^{[\tau_i,\underline{\mathscr{R}}_i^{\mathrm{l}}|\mathscr{R}_i^{\mathrm{r}}]} \underline{I}_i'^{\mathrm{l}} \,|\, I_i'^{\mathrm{r}}$ are obtained from $I_i^{\mathrm{l}} \rightarrow^{[\lambda_i,\mathscr{R}_i^{\mathrm{l}}]} I_i'^{\mathrm{l}}$ and $\underline{I}_i^{\mathrm{l}} \rightarrow^{[\underline{\lambda}_i,\underline{\mathscr{R}}_i^{\mathrm{l}}]} \underline{I}_i'^{\mathrm{l}}$ as done above. Since computations ξ_0^{l} and ξ_1^{l} have the *same* label h and are sewed with ξ_0^{l} in the *same* manner, the recomposed computation ξ_1 originates the *same* partial ordering derivation of ξ_0.

When there exist quadruples with $\underline{\lambda}_i \neq (\lambda_i^-)^-$, we have to rearrange computation ξ_1^{l} in order to go back to the previous case, and "put Humpty together again" properly. This can always be done, since these quadruples generate the corresponding events in different total orderings, and thus these events are *concurrent.* Thus, we can apply Theorem 4.6 to switch transitions in ξ_1^1 and still obtain a legal computation originating the same partial ordering derivation with label h^1. □

References

[1] D. Austry and G. Boudol, Algèbre de processus et synchronization, *Theoret. Comput. Sci.* **30** (1984) 91-131.
[2] G. Boudol and I. Castellani, On the semantics of concurrency: partial orders and transition systems, *Theoret. Comput. Sci.* **59** (1988) 25-84.
[3] S.D.Brookes, C.A.R. Hoare and A.D. Roscoe, A theory of communicating sequential processes, *J. ACM* **31** (1984) 560-599.
[4] J.A. Bergstra and J.-W. Klop, Process algebra for synchronous communication, *Inform. and Control* **60** (1984) 109-137.
[5] M. Broy and T. Steicher, Views of distributed systems, in M. Venturini Zilli, ed., *Proc. Advanced School on Mathematical Models for the Semantics of Parallelism*, Lecture Notes in Computer Science **280** (Springer, Berlin, 1987) 114-143.
[6] I. Castellani and M. Hennessy, Distributed bisimulations, *J. ACM* **36** (1989) 887-911.
[7] Ph. Darondeau and P. Degano, Causal trees, in: G. Ausiello et al., eds., *Proc. 16th ICALP*, Lecture Notes in Computer Science **372** (Springer, Berlin, 1989) 234-248.
[8] P. Degano, R. De Nicola and U. Montanari, Partial ordering derivations for CCS, in: *Proc. FCT*, Lecture Notes in Computer Science **199** (Springer, Berlin, 1985) 520-533.
[9] P. Degano, R. De Nicola and U. Montanari, Observational equivalences for concurrency models, in: M. Wirsing, ed., *Formal Description of Programming Concepts III* (North-Holland, Amsterdam, 1987) 105-132.
[10] P. Degano, R. De Nicola and U. Montanari, CCS is an (augmented) contact-free C/E system, in: M. Venturini Zilli, ed., *Proc. Advanced School on Mathematical Models for the Semantics of Parallelism*, Lecture Notes in Computer Science **280** (Springer, Berlin, 1987) 144-165.
[11] P. Degano, R. De Nicola and U. Montanari, A distributed operational semantics for CCS based on condition/event systems, *Acta Inform.* **26** (1988) 59-91.
[12] P. Degano, R. De Nicola and U. Montanari, On the consistency of truly concurrent operational and denotational semantics, in: *Proc. LICS '88*, Edinburgh (1988) 133-141.
[13] F. De Cindio, G. De Michelis, L. Pomello and C. Simone, Milner's communicating systems and Petri nets, in: *Selected papers of 3rd European Workshop on Applications and Theory of Petri Nets, IF* **66** (Springer-Verlag, Heidelberg, 1983) 40-59.
[14] P. Degano, R. Gorrieri and S. Marchetti, An exercise in concurrency: a CSP process as a condition/event system, in: G. Rozenberg, ed., *Advances in Petri Nets 1988*, Lecture Notes in Computer Science **340** (Springer, Berlin, 1988) 85-105.
[15] R. De Nicola and M. Hennessy, Testing equivalence for processes, *Theoret. Comput. Sci.* **34** (1984) 83-133.
[16] Ph. Darondeau and L. Kott, On the observational semantics of fair parallelism, in: J. Diaz, ed., *Proc. 10th ICALP*, Lecture Notes in Computer Science **154** (Springer, Berlin, 1983) 147-151.
[17] P. Degano and U. Montanari, A model of distributed systems based on graph rewriting, *J. ACM* **34** (1987) 411-449.
[18] P. Degano and U. Montanari, Concurrent histories: a basis for observing distributed systems, *J. Comput. System Sci.* **34** (1987) 442-461,
[19] U. Goltz and A. Mycroft, On the relationships of CCS and Petri nets, in: J. Paredaens, ed., *Proc. 11th ICALP*, Lecture Notes in Computer Science **172** (Springer, Berlin, 1984) 196-208.
[20] U. Goltz and W. Reisig, The non-sequential behaviour of Petri nets, *Inform. and Control* **57** (1983) 125-147.
[21] R. van Glabbeek and F. Vaandrager, Petri net models for algebraic theories of concurrency, in: J. W. de Bakker, A.J. Nijman and P.C. Treleaven, eds., *Proc. PARLE Conf.*, Lecture Notes in Computer Science **259** (Springer, Berlin, 1987) 224-242.

[22] M. Hennessy, *An Algebraic Theory of Processes* (MIT Press, Cambridge, MA, 1988).

[23] M. Hennessy and R. Milner, Algebraic laws for nondeterminism and concurrency, *J. ACM* **32** (1985) 137-161.

[24] L. Lamport, Time, clocks and the ordering of events in a distributed system, *Comm. ACM* **12** (1978) 558-564.

[25] L. Lamport, What good is temporal logic?, in: R.E.A. Mason, ed., *Proc. IFIP Congress '83* (North-Holland, Amsterdam, 1983) 657-668.

[26] A. Mazurkiewicz, Concurrent program schemas and their interpretation, in: *Proc. Aarhus Workshop on Verification of Parallel Programs*, 1977.

[27] R. Milner, *A Calculus of Communicating Systems*, Lecture Notes in Computer Science **92** (Springer, Berlin, 1980).

[28] R. Milner, Calculi for synchrony and asynchrony, *Theoret. Comput. Sci.* **25** (1983) 267-310.

[29] R. Milner, Lectures on a calculus for communicating systems, in: M. Broy, ed., *Control Flow and Data Flow: Concepts of Distributed Programming*, NATO ASI Series F, **14** (Springer, Berlin, 1985) 205-228.

[30] G. Milne, CIRCAL and the representation of communication, concurrency and time, *ACM TOPLAS* 7 (1985) 270-298.

[31] M. Nielsen, G. Plotkin and G. Winskel, Petri nets, event structures and domains, Part 1, *Theoret. Comput. Sci.* **13** (1981) 85-108.

[32] M. Nivat, Behaviours of processes and synchronized systems of processes, in: M. Broy and G. Schmidt, eds., *Theoretical Foundations of Programming Methodology* (Reidel, Dordrecht, 1982) 473-550.

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

[34] E.-R. Olderog, Operational Petri net semantics for CCSP, in: G. Rozenberg, ed., *Advances in Petri Nets 1987*, Lecture Notes in Computer Science **266** (Springer, Berlin, 1987) 196-223.

[35] D. Park, Concurrency and automata on infinite sequences, in: *Proc. GI*, Lecture Notes in Computer Science **104** (Springer, Berlin, 1981) 167-183.

[36] C.A. Petri, Concurrency, in: *Net Theory and Applications*, Lecture Notes in Computer Science **84** (Springer, Berlin, 1980) 1-19.

[37] G. Plotkin, A structural apporach to operational semantics, Technical Report DAIMI FN-19, Aarhus University, Department of Computer Science, Aarhus, 1981.

[38] V. Pratt, Modelling concurrency with partial orders, *Internat. J. Parallel Programming* **15** (1986) 33-71.

[39] M.W. Shields, Non-sequential behaviour I, Internal Report CSR-120-82, Department of Computer Science, University of Edinburgh, 1982.

[40] J. Winkowski, Behaviours of concurrent systems, *Theoret. Comput. Sci.* **12** (1980) 39-60.

[41] G. Winskel, Event structures for CCS and related languages, in: M. Nielsen and E.M. Schmidt, eds., *Proc. 9th ICALP*, Lecture Notes in Computer Science **140** (Springer, Berlin, 1982) 561-576.

[42] G. Winskel, Petri nets, algebras, morphisms and compositionality, *Inform. and Control* **72** (1987) 197-238.

[43] A Rabinovich and B.A. Trakhtenbrot, Behavior structures and nets, *Fund. Inform.* **XI**(4) (1988) 357-403.

[44] G. Boudol and I. Castellani, A non-interleaving semantics for CCS based on proved transitions. *Fund. Inform.* **XI**(4) (1988) 433-452.