An Extended Expansion Theorem

Gian Luigi Ferrari†, Roberto Gorrieri‡, and Ugo Montanari‡

†IEI-CNR, Via Santa Maria 46, I-56125 Pisa Italy

‡Dipartimento di Informatica, Università di Pisa,
Corso Italia 40, I-56125 Pisa, Italy

Abstract

Closed CCS (CCCS) is a CCS-like algebra of processes with a generalized form of prefixing based on a full-fledged algebra of transitions rather than on basic actions only. The basic idea is that the generalized prefixing operator takes a transition t, or rather its observation ω, a process E and yields the process $t.E$. From an operational standpoint, the process $t.E$ may evolve to E by performing a transition labelled by ω. By exploiting the algebra of transitions, we define a general form of *expansion theorem* which is the heart of a finite axiomatization of a strong observational equivalence for finite CCCS agents. By adding the axioms concerning the interpretation of the operations of the algebra of observations, we still obtain a sound and complete axiomatization of the corresponding bisimulation equivalence. For instance, it is possible to define the classical expansion theorem, or versions of it which handle partial ordering based observations.

1 Introduction

Many different models have been proposed to specify the behaviour of concurrent and distributed systems. We single out two approaches: the *interleaving* approach ([Mil80,Mil89], [BHR84,Ho85], [AB84], [BK84], [He88]) and the *true concurrency* approach ([Re85], [NPW81], [Pr86], [BC88], [DM87,DDM88a,DDM90a], [Ol87], [RT88]).

The debate and the arguments between the supporters of either these antagonist approaches have not yet led to a clear measure of superiority of one over the other. In our view, the main merit of the former is its well-established theory, whilst the latter gives an intuitively more convincing representation of concurrent system behaviour. This paper aims at giving a contribution in filling the gap between the two approaches by providing also the latter with an axiomatic theory which can be naturally specialized to the interleaving case.

The development of the interleaving approach to concurrency is well illustrated by Milner's work on CCS [Mil80,Mil89]. Considering concurrent systems as structured entities, which interact by some synchronization mechanisms, naturally leads to the definition of operations for building new systems from existing ones: every system can be seen as a term of the free algebra over this set of operations. The resulting *process description language* (the one proposed by Milner is called *Calculus of Communicating Systems (CCS)*) comes equipped with an operational semantics which takes the form of a *labelled transition system* [Plo81]. However, these operational descriptions are too intensional. More abstract semantics of the language are obtained by introducing behavioural equivalences which identify process terms which exhibit the same behaviour in accordance with certain observational scenarios.

Many of these behavioural equivalences are based on the notion of *bisimulation* [Par81]. Several bisimulation based equivalences are completely characterized by equational laws between process terms; in other words, the equivalences are actually congruences obtained by making the quotient of the free algebra

*Work partially supported by ESPRIT Basic Research Action 3011, CEDISYS, and by Progetto Finalizzato Informatica e Calcolo Parallelo, obiettivo LAMBRUSCO

with respect to the equational laws. For finite CCS (i.e., CCS without recursion), the so called *strong bisimulation equivalence* is characterized [HM85] by a finite set of axioms plus an axiom schema (thus infinitary in principle): the *expansion theorem*. These axioms directly state that all the CCS operators can be defined as derived operations in terms of prefixing and summation. In particular, the expansion theorem explains how parallel composition can be stepwise reduced to sequentiality and nondeterminism. As an instance of this law, the CCS process terms $\alpha.nil \mid \beta.nil$ and $\alpha.\beta.nil + \beta.\alpha.nil$ are identified. The intuitive interpretation of this identification is that the parallel execution of the actions α and β is represented by the non deterministic choice of their interleavings. Indeed, this is the essence of interleaving semantics.

Recently, Moller [Mol90a,Mol90b] proved that a finite axiomatization for bisimulation based equivalences can be obtained only at the price of introducing auxiliary operators like Bergstra and Klop's *left, right* and *communication merge* [BK84,BK85]: the expansion theorem is thus replaced by a finite set of axioms relating parallel composition to the auxiliary operators, and showing how the auxiliary operators behave with respect to the two fundamental operators of prefixing and summation.

Truly concurrent models describe the behaviour of systems by means of partial orderings where the ordering relation mirrors the causal dependencies, and concurrency is represented as absence of ordering. Even if this approach provides a more faithful account of distributed computations, it lacks a satisfactory algebraic treatment. Indeed, the techniques for defining and handling partial orders in an algebraic style still miss conclusive achievements. As a consequence, results on equational theories for truly concurrent models are not firm yet, and the interleaving approach, even if less powerful, has been adopted in giving the semantics of process description languages.

The key problem in concurrency theory is to understand which are the mechanisms to determine the observable behaviours of concurrent and distributed computations. In fact, most of the behavioural equivalences can be understood in terms of the assumptions on the observation mechanism: two processes are equivalent provided that no observations can tell them apart. Partial orderings of events give a great power in modelling process behaviour, however the definition of an observational scenario for true concurrency seems too difficult to formalize. An argument in favour of this thesis is the lack of axiomatizations for truly concurrent behavioural equivalences.

An attempt to provide equational characterizations of observational scenarios for true concurrency has been recently given by Degano, De Nicola and Montanari [DDM91]. Their observational framework uses certain node-labelled trees, called *Observation Trees*, which are derived from *Nondeterministic Measurement Systems* introduced in [DDM87]. The idea is that the nodes of an observation tree represent computations, and the labels give their observations (taken from a certain domain of observations, possibly based on partial orderings). On this class of trees, several bisimulation equivalences, parameterized with respect to the domain of the observations, can be defined. It turns out that (finite) observation trees can be described syntactically, and, moreover, complete sets of axioms can be provided for the various equivalences. Such axiomatizations are independent from the domain of observations actually chosen.

However, by considering computations (rather than transitions) and their observations, such trees give an *integral* description of behaviours. Furthermore, the axiomatizations are not in the algebra of processes but rather in the algebra of observation trees. We have already remarked the important role played by the expansion theorem in reducing all operators to sum and prefixing. If we would be able to exhibit an expansion theorem also for true concurrency, then we would obtain a firm ground to provide axiomatizations on the algebra of processes also handling truly concurrent observational scenarios.

This paper aims at solving this problem by providing a general form of expansion theorem which covers the true concurrent case, besides the obvious interleaving one.

Let us start with a short discussion aiming at understanding the reasons which makes parallel composition a derived operator in the interleaving semantics. Basically, this is due to the fortunate coincidence that elementary actions are both the building blocks syntactically generating process terms *via* prefixing, and the observations out of a transition. In this way, the behaviour of an agent can be represented as the tree[1] obtained by unfolding the transition system, and, equivalently, as a term of the language over the restricted sequential/nondeterministic signature.

[1] A notable example is provided by Milner's Synchronization Trees.

The identity between observations and prefixed actions is indeed the basic assumption which makes the expansion theorem definable (as we will see later, also another, more subtle condition is essential).

For a long time in the true concurrency side there was a rather discouraging feeling about the existence of any axiomatization of bisimulation based equivalences, also motivated by the works of Grabowski and Gisher ([Gr81], [G88]). They proved that no finite syntax does exist for general *pomsets* (acronym for partially ordered multisets), and therefore that no algebraic characterization of them can be given except for the simple case of series-parallel pomsets. Indeed, Boudol and Castellani [BC88] proved that an expansion theorem is definable within the (finite) algebra of series-parallel pomsets. As they pointed out, the criticism for the absence of an axiomatization for causality based bisimulation equivalences "should not be moved against the expansion theorem, but rather against the lack of structure in the actions of transition systems".

In this perspective, an important breakthrough has been realized with the introduction of *Causal Trees* [DD89,DD90], which are trees with a richer labelling than Synchronization Trees. Besides the action μ performed, an arc is also labelled by a set K of integers playing the role of backward pointers to the transitions which caused the present arc. Causal Trees supply a concrete representation of Nondeterministic Measurament Systems with mixed ordering observations [DDM89], and have to be properly understood as executions of Event Structures [NPW81]. Darondeau and Degano were able to define an expansion theorem which fully expresses the intended causal dependencies among actions performed by CCS process terms, and, consequently, an axiomatization for *causal* bisimulation (an alternative, yet equivalent formulation of the more popular *history preserving* bisimulation [DDM89], [RT88], [GG89]). However, this expansion theorem, besides making use of some auxiliary operators, exploits the richer observations. Indeed, a causal tree can be seen as a term of an algebra with the usual two operations of prefixing and summation, but where prefixing is actually defined on observations of the form (μ,K). As a consequence, even if parallel composition can be reduced to nondeterminism and sequentiality, a parallel CCS term cannot be reduced to a *sequential* CCS term through the expansion theorem, because labels of the form (μ,K) are not arguments of the prefixing operator. Again, this amounts to saying that the axiomatization given in [DD89] is on the algebra of Causal Trees and not on the algebra of CCS process terms. Furthermore, each label such as (μ, K) has not a meaning *in se* (at least when $K \neq \emptyset$), in the sense that the actual meaning of the set K needs the past history to be properly defined. In other words, the mechanism of observing partial orderings offered by Causal Trees is not satisfactory since it is not completely incremental.

The crucial point is that models based directly on partial orderings give too abstract representations of process behaviours, on which a definition of sequential composition cannot be nicely defined (except for the trivial case in which all the events of the former process precede all the events of the latter). Indeed, we conjecture that an appealing incremental description of truly concurrent computations cannot be given without resorting to more concrete observation structures like *Concurrent Histories* [DM87,DDM90a], where partial orderings are enriched with origin and destination processes, or *Concatenable Processes* [DMM89].

From the above discussion, it should be clear that in order to get an expansion theorem for true concurrency we need to include truly concurrent observations as argument of the prefixing operator (not in the semantic model, like for Observation Trees and Causal Trees, but in the syntax of the calculus itself). To this aim we equip prefixed actions with an algebraic structure, thus obtaining an algebra of observations in the vein of ([MY89], [FM90], [GM90]). The algebraic structure on observations is not necessarily free since it may reflect a particular interpretation of the operations. There is a great variety of possibilities as to the choice of the interpretation of the operations. As an example, the classical interleaving approach is easily recovered by interpreting the operations to yield elementary actions. Moreover, also a truly concurrent interpretation is naturally obtained.

The technical development is the following. In CCS there is a prefix operator which from the atomic action μ and the agent E yields the process $\mu.E$. In CCCS, we allow a generalized prefixing operation which takes a transition t, an agent E and yields the process $t.E$.

The operational semantics of CCCS is given by axioms and inference rules. However, in CCCS the operational derivations are not statements of the form $E_1 \xrightarrow{\omega} E_2$, but rather assertions like $t : E_1 \xrightarrow{\omega} E_2$ meaning that E_1 evolves to E_2 via a transition t labelled by ω.

In our approach, the CCS axiom $\mu.E \xrightarrow{\mu} E$, or equivalently the assertion $[\mu, E>: \mu.E \xrightarrow{\mu} E$ where $[\mu, E>$ is the syntactic term (i.e., the name: see e.g. [FM90]) for the transition $\mu.E \xrightarrow{\mu} E$, is replaced by an axiom and an inference rule:

$$\mu : \mu.nil \xrightarrow{\mu} nil$$

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{[t, E\!>: t.E \xrightarrow{\omega} E}$$

where ω denotes the observation out of the transitions t and $[t, E>$.

It remains to be specified which is the observation ω out of any transition t. At first sight, we could say that the most concrete observation is the transition t itself. In this way the operators of the algebra of transitions describe the context information on the observation itself. For instance, one of the rules for parallel composition is as follows:

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{t\rfloor E_3 : E_1 \mid E_3 \xrightarrow{\omega\rfloor -} E_2 \mid E_3}$$

where $\omega\rfloor -$ denotes that the observation ω is in a context with some idle subsystem at right. In this way, the observation is enriched with a topological information stating where the action takes place, i.e., its *locality*. Notice that the operation on the arrow does not take the agent E_3 as an operand. In fact, while in principle a development taking into account a more informative observation might be possible, we considered including agents in the observations as incompatible with the message-passing philosophy of CCS.

We have followed this approach, except for two cases. The first case has been already presented in the rule for generalized prefixing $[t, E>$ and is rather natural: the observation of a transition $[t, E>$ is again the observation of the transition t. Intuitively, this means that the operation does change neither the action, nor its locality. More subtle is the other case, concerning nondeterminism. On one hand, it seems reasonable to enrich the observation also with the information on the rejected context:

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{t <\!+E_3 : E_1 + E_3 \xrightarrow{\omega <+-} E_2}$$

On the other hand, a choice context does not change the locality of an action (we could say that $\omega <+-$ reveals the locality *after* the choice), and thus it can be safely omitted in the observation:

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{t <\!+E_3 : E_1 + E_3 \xrightarrow{\omega} E_2}.$$

This concludes the description of the operational semantics of CCCS. To obtain more abstract semantics we consider the notion of bisimulation. Two agents are equivalent if they are able to perform transitions with the same observations, evolving to equivalent agents. For instance, the agents

$$\alpha.nil \qquad\qquad \alpha.nil + \alpha.nil$$

are bisimilar. Instead, the agents

$$\alpha.nil \mid \beta.nil \qquad\qquad \alpha.\beta.nil + \beta.\alpha.nil$$

are not equivalent. The first agent may perform the transition $\alpha\rfloor\beta.nil$ with observation $\alpha\rfloor -$ while the second agent cannot. This simple example amounts to saying that, at least when operations on observations are not interpreted, concurrency is not simulated in terms of nondeterminism.

In this paper, we prove that the bisimulation equivalence is characterized by a complete set of axioms. The axiomatization does not depend on the actual algebra of observations. As a consequence, by suitably interpreting the operations on observations to yield *specific* observations we still obtain sound and complete axiomatizations of the corresponding bisimulation equivalences. For instance, the axiomatization of Milner's Strong Observational Equivalence can be obtained by interpreting the operations to yield actions.

The heart of the axiomatization is given by the following axiom which states the expansion theorem.

$$E_1 \mid E_2 = E_1\rfloor E_2 + E_1\lfloor E_2 + E_1 \parallel E_2$$

The axiom states that the operator of parallel composition is expressed in terms of some auxiliary operators: left merge $\rfloor$, right merge $\lfloor$ and synchronization merge $\|$. Moreover, there are axioms stating the behaviour of the auxiliary operators with respect to prefixing and summation. In particular, the following axioms ensure that, in breaking down the parallel operator, the information on causal dependencies are maintained through the operation of the generalized prefixing.

$$t.E_1 \rfloor E_2 = (t \rfloor E_2).(E_1 \mid E_2)$$

$$E_2 \lfloor t.E_1 = (E_2 \lfloor t).(E_2 \mid E_1)$$

$$t_1.E_1 \parallel t_2.E_2 = (t_1 \mid t_2).(E_1 \mid E_2)$$

Other axioms state how the auxiliary operators distribute with respect to the operator of non deterministic choice. E.g.:

$$(E_1 + E_2) \rfloor E_3 = E_1 \rfloor E_3 + E_2 \rfloor E_3$$

Finally, there are axioms stating that the auxiliary operators can be eliminated from process expressions; thus, their role is indeed auxiliar.

$$nil \rfloor E = E \lfloor nil = E \parallel nil = nil \parallel E = nil$$

A further set of axioms is given to eliminate the restriction and the relabelling operators.

As an example of application of these axioms, let us consider the agent $t.E_1 \mid E_2$. In order axiom

$$t.E_1 \mid E_2 = t.E_1 \rfloor E_2 + t.E_1 \lfloor E_2 + t.E_1 \parallel E_2$$

to be sound, the two transitions

$$[t, E_1 > \rfloor E_2$$

from the left member, and (since $t.E_1 \rfloor E_2 = (t \rfloor E_2).E_1 \mid E_2$)

$$[t \rfloor E_2, E_1 \mid E_2 >< +(t.E_1 \lfloor E_2 + t.E_1 \parallel E_2)$$

from the right member, must have the same observation. This is the case, since, as discussed above, both have the same label as $t \rfloor E_2$. This example clarifies that the essence of the expansion theorem relies not only upon the fact that prefixing must be defined on transitions (keeping the same observation), but also that the choices made inside a transition must not be observed.

In Section 2 we present an *absolute* version of $CCCS$, where the operations on the observations are not interpreted, and where the prefix operator contains a transition. In Section 3 we give our complete axiomatization of the strong observational equivalence. In Section 4 we present a *parameterized* version $CCCS_\Omega$, where Ω is an algebra of observations and where the prefix operator contains an observation. Essentially the same axiomatization (but now enriched with axioms for Ω) holds and is complete for $CCCS_\Omega$. Finally, in Section 5 we consider two particular observation algebras: $\mathcal{I}$, yielding the classical version of CCS, and $\mathcal{SH}$, the algebra of *Spatial Histories*, for a true concurrency version. Spatial Histories are characterized (with respect e.g. to Concatenable Petri Processes [DMM89], Concatenable Concurrent Histories [FMM91a] and Causal Streams [FMM91b]) by the fact that they are not ranked, i.e. the operation of sequential composition is always defined, and is performed via a simple notion of unification. In the conclusion, we point out that just adding this operation of sequentialization to the algebra of transitions would allow us to extend our results to handle atomic actions.

2 Closed CCS: Operational and Bisimulation Semantics

In this section we introduce Closed CCS. We assume some familiarity with CCS. Let Δ (ranged over by α) be the alphabet of basic actions, and $\overline{\Delta}$ the alphabet of complementary actions ($\Delta = \overline{\overline{\Delta}}$). The set $\Lambda = \Delta \cup \overline{\Delta}$ will be ranged over by λ. Furthermore, as in CCS we use the special action τ, $\tau \notin \Lambda$ for internal moves, and moreover, we adopt the special action $\delta \notin (\Lambda \cup \{\tau\})$ for error. Finally, let $A = \Lambda \cup \{\tau, \delta\}$ (ranged over by μ) be the set of elementary actions.

The standard operational semantics of process algebras is given by axioms and inference rules [Plo81] which allow us to derive statements of the form $E_1 \xrightarrow{\omega} E_2$, where ω is the observation associated to the derivation. Here, instead, we write

$$t : E_1 \xrightarrow{\omega} E_2$$

to indicate that t is the name (proof) of a transition from E_1 to E_2 with observation ω.

Definition 1 *(Closed CCS: Absolute Form)*
A CCCS agent E has the following syntax:

$$E ::= nil\,,\ t.E\,,\ E\backslash\alpha\,,\ E[\Phi]\,,\ E_1 + E_2\,,\ E_1 \mid E_2$$

where t is a transition, *and Φ is a permutation on the set of elementary actions, fixing τ, δ, and the operation of complementation.*
The formation rules of the transitions are given as follows.

$$\mu : \mu.nil \xrightarrow{\mu} nil$$

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{[t, E>: t.E \xrightarrow{\omega} E}$$

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{t\backslash\alpha : E_1\backslash\alpha \xrightarrow{\omega\backslash\alpha} E_2\backslash\alpha}$$

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{t[\Phi] : E_1[\Phi] \xrightarrow{\omega[\Phi]} E_2[\Phi]}$$

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{t <\!+E : E_1 + E \xrightarrow{\omega} E_2} \qquad \frac{t : E_1 \xrightarrow{\omega} E_2}{E\!+\!> t : E + E_1 \xrightarrow{\omega} E_2}$$

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{t\rfloor E : E_1 \mid E \xrightarrow{\omega\rfloor -} E_2 \mid E} \qquad \frac{t : E_1 \xrightarrow{\omega} E_2}{E\lfloor t : E \mid E_1 \xrightarrow{-\lfloor\omega} E \mid E_2}$$

$$\frac{t : E_1 \xrightarrow{\omega} E_2,\ t' : E_1' \xrightarrow{\omega'} E_2'}{t \mid t' : E_1 \mid E_1' \xrightarrow{\omega\mid\omega'} E_2 \mid E_2'}$$

□

We can comment briefly on the definition of the transitions and their observations ω. As we have already remarked, the transitions have to be interpreted as proofs of the operational derivations. Indeed, there is an operator for each rule. This is the standard construction of the *proved transition system* [DDM85,DDM90b], [BC89], [MY89], [FM90], [BC90]. Computations are obtained just by concatenating transitions.

Example 2 *(Operational Semantics)*
Let $E = \beta.\alpha.nil \mid \gamma.nil$ be a CCCS term. To illustrate how the operational semantics of CCCS works, we show a complete derivation for agent E. Applying the inference rules we have:

$$\dfrac{\dfrac{\beta : \beta.nil \xrightarrow{\beta} nil}{[\beta, \alpha.nil>: \beta.\alpha.nil \xrightarrow{\beta} \alpha.nil}}{[\beta, \alpha.nil>]\gamma.nil : \beta.\alpha.nil \mid \gamma.nil \xrightarrow{\beta]-} \alpha.nil \mid \gamma.nil}$$

Let us consider the term $\delta.nil \mid \alpha.nil$. Applying the inference rules we have:

$$\dfrac{\delta : \delta.nil \xrightarrow{\delta} nil, \quad \alpha : \alpha.nil \xrightarrow{\alpha} nil}{\delta \mid \alpha : \delta.nil \mid \alpha.nil \xrightarrow{\delta|\alpha} nil \mid nil}$$

□

The last example above shows that the action δ may appear as argument of the operations of the observations ω. Recall that the action δ has the meaning of indicating the occurrence of an error. Thus, we need to determine which transitions should be considered *correct* or *proper*.

Definition 3 *(Proper Transitions and Observations)*
A transition $t : E_1 \xrightarrow{\omega} E_2$ is called an error transition *provided that either:*

- $t = \delta$, *or*
- $t = \sigma(t')$ *for some error transition t' and operation σ.*

A transition $t : E_1 \xrightarrow{\omega} E_2$ is called a proper transition *(observation ω is called a* proper observation*) provided that t is not an error transition.* □

We now introduce the bisimulation semantics of Closed CCS. Only proper transitions and observations are involved in the definition of bisimulation, i.e. only proper transitions can distinguish between CCCS terms.

Definition 4 *(Strong Bisimulation)*
A binary symmetric relation R on Closed CCS terms is called bisimulation *if and only if R satisfies the following clause:*

- *if $u_1 \; R \; u_2$, and $t_1 : u_1 \xrightarrow{\omega} v_1$ is a proper transition, then there exists t_2 such that $t_2 : u_2 \xrightarrow{\omega} v_2$ is a proper transition and $v_1 \; R \; v_2$.*

Two Closed CCS terms, u_1 and u_2, are bisimilar, and we write $u_1 \sim u_2$, if there exists a bisimulation relating them.□

It is a well known fact [Par81] that the arbitrary union of bisimulation relations is again a bisimulation.

Proposition 5 *(Strong Observational Equivalence)*
Relation $\sim$ is the maximal bisimulation and it is an equivalence relation. It is called Strong Observational Equivalence. □

It is easy to show that the equivalence relation $\sim$ is indeed a congruence, i.e. equivalent agents inserted within any context are still equivalent.

Theorem 6 *(Congruence)*
The relation $\sim$ is a congruence
Proof. *Standard. See, for instance the proof given in [Mil89] for CCS strong observational congruence.* □

Example 7 *(Some Equivalent CCCS Processes)*
Let E be a CCCS process. The process $\delta.E$ is equivalent to nil: there is no proper transition which can discriminate between the two processes.
It is easy to see that the processes $E + E$ and E are equivalent. Moreover, $E + nil$ is equivalent to E. The process $(t\rfloor nil).(E \mid nil)$ is equivalent to $t.E \mid nil$. However, it is not true that $(t\rfloor E_2).(E_1 \mid E_2)$ is equivalent to $t.E_1 \mid E_2$, for any CCCS term E_1 and E_2. For instance, take $(\alpha\rfloor\beta.nil).(nil \mid \beta.nil)$ and $\alpha.nil \mid \beta.nil$. The second process can perform a transition with observation $\alpha \mid \beta$ while the first process cannot.
Finally, the process $\alpha.nil \mid \beta.nil$ is not equivalent to the process $\beta.nil \mid \alpha.nil$. The first process can perform a transition with observation $\alpha\rfloor-$ while the second cannot. This final example expresses that the equivalence $\sim$ discriminates between processes which have a different physical distribution, in other words we observe the localities of the composing subagents. □

Example 8 *(Expressive Power)*
The two CCCS processes $E_1 = \alpha.(\beta.nil+\gamma.nil)+\alpha.nil \mid \beta.nil$, $E_2 = \alpha.(\beta.nil+\gamma.nil)+\alpha.nil \mid \beta.nil+\alpha.\beta.nil$ are indistinguishable by Pomset Bisimulation Equivalence [BC88]. However, they are distinguished by $\sim$. This is because the process E_2 can perform a transition with observation α ending in a state from which a transition with observation γ is impossible. The process E_1 cannot do this because there is no transition with observation α from the state $\alpha.nil \mid \beta.nil$, although there is a transition with observation $\alpha\rfloor-$. □

3 Finite Complete Axiomatization

In this section we introduce an equational theory $\mathcal{F}$ which characterizes the bisimulation equivalence $\sim$ introduced in the previous section. To this aim we introduce some auxiliary operators, following Bergstra and Klop Algebra of Communicating Processes (ACP) [BK84,BK85].

Definition 9 *(The Equational Theory $\mathcal{F}$)*
The set of terms of the equational theory $\mathcal{F}$ is given by the following grammar:

$$F ::= nil\ ,\ t.F\ ,\ F\backslash\alpha\ ,\ F[\Phi]\ ,\ F_1 + F_2\ ,\ F_1 \mid F_2\ ,\ F_1\rfloor F_2\ ,\ F_1\lfloor F_2\ ,\ F_1 \parallel F_2$$

where t is a transition *of Closed CCS.*
The equational theory $\mathcal{F}$ is given as follows:

$S1$ $\quad F + F = F$

$S2$ $\quad F_1 + F_2 = F_2 + F_1$

$S3$ $\quad F_1 + (F_2 + F_3) = (F_1 + F_2) + F_3$

$S4$ $\quad F + nil = F$

N $\quad t.E = nil$, where t is an error transition

T $\quad \dfrac{t : E_1 \xrightarrow{\omega} E_2, t' : E_1' \xrightarrow{\omega} E_2'}{t.E = t'.E}$

EX $\quad F_1 \mid F_2 = F_1\rfloor F_2 + F_1\lfloor F_2 + F_1 \parallel F_2$

$D1 \quad t.F_1 \rfloor F_2 = (t \rfloor F_2).(F_1 \mid F_2)$

$D2 \quad F_2 \lfloor t.F_1 = (F_2 \lfloor t).(F_2 \mid F_1)$

$D3 \quad t_1.F_1 \parallel t_2.F_2 = (t_1 \mid t_2).(F_1 \mid F_2)$

$D4 \quad (F_1 + F_2) \rfloor F = F_1 \rfloor F + F_2 \rfloor F$

$D5 \quad F \lfloor (F_1 + F_2) = F \lfloor F_1 + F \lfloor F_2$

$D6 \quad (F_1 + F_2) \parallel F = F_1 \parallel F + F_2 \parallel F$

$D7 \quad F \parallel (F_1 + F_2) = F \parallel F_1 + F \parallel F_2$

$R1 \quad t.F \backslash \alpha = t \backslash \alpha . F \backslash \alpha$

$R2 \quad t.F[\Phi] = t[\Phi].F[\Phi]$

$R3 \quad (F_1 + F_2) \backslash \alpha = F_1 \backslash \alpha + F_2 \backslash \alpha$

$R4 \quad (F_1 + F_2)[\Phi] = F_1[\Phi] + F_2[\Phi]$

$Z \quad nil \backslash \alpha = nil[\Phi] = nil \mid nil = nil \rfloor F = F \lfloor nil = F \parallel nil = nil \parallel F = nil$

□

We can comment briefly on the definition of the equational system. In the line of ACP, three operators $\rfloor$, right merge, $\lfloor$, left merge and $\parallel$, synchronization merge have been introduced. The interpretation of the term $F_1 \rfloor F_2$ is that the process F_1 has more priority than F_2: the interpreter is committed to choose the first transition among those F_1 can perform. The dual phenomenon happens with the process $F_1 \lfloor F_2$. The process $F_1 \parallel F_2$ is committed to a synchronization.

The axioms (S1 - S4) express that (F, +, nil) is an *abelian monoid.* Notice that these are Milner's axioms for the Strong Observational Congruence. The axiom (N) expresses that only proper observations can discriminate the behaviour of agents. Notice that we do not make any distinction between a legal termination, represented by *nil*, and a termination due to the occurrence of an error transition. The axiom (T) expresses that we identify processes prefixed with transitions which give rise to the same observation. This axiom amounts to saying that transitions really stand for observations. The axiom (EX) states the expansion theorem. A parallel process $F_1 \mid F_2$ can proceed either by choosing to perform a step in F_1, or in F_2, or to synchronize the two processes.

The axioms (D1 - D7) state the distribution of the auxiliary operators with respect to the fundamental operators of sum and prefixing. The meaning of the remaining axioms is immediate.

Example 10 *(Expansion of Parallel CCCS Agents)*
Let $\alpha.nil \mid \beta.nil$ be a CCCS agent. By applying the laws of the equational system $\mathcal{F}$ as rewriting rules we can infer the equality

$$\alpha.nil \mid \beta.nil = \alpha.nil \rfloor \beta.nil + \alpha.nil \lfloor \beta.nil + \alpha.nil \parallel \beta.nil$$

$$= (\alpha \rfloor \beta.nil).(nil \mid \beta.nil) + (\alpha.nil \lfloor \beta).(\alpha.nil \mid nil) + (\alpha \mid \beta).(nil \mid nil)$$

$$= (\alpha \rfloor \beta.nil).(nil \lfloor \beta).(nil \mid nil) + (\alpha.nil \lfloor \beta).(\alpha \rfloor nil).(nil \mid nil) + (\alpha \mid \beta).(nil \mid nil)$$

$$= (\alpha \rfloor \beta.nil).(nil \lfloor \beta).nil + (\alpha.nil \lfloor \beta).(\alpha \rfloor nil).nil + (\alpha \mid \beta).nil$$

□

We will prove that the equational theory above is a characterization of the bisimulation equivalence $\sim$. It is worth-noting that axiom (T) points out that prefixing with transitions stands for prefixing with observations. To illustrate this fact we have the following proposition.

Proposition 11 *(Prefixing Theorems)*
Let t be a transition, and E_1, E_2 be CCCS agents. The following equalities hold in $\mathcal{F}$

$$t.E_1 = [t, E_2{>}.E_1$$

$$(t <{+}E_1).E_2 = (E_1{+}> t).E_2 = t.E_2$$

$$(t\rfloor E_1).E_2 = (t\rfloor E_1').E_2$$

$$(E_1\lfloor t).E_2 = (E_1'\lfloor t).E_2$$

Proof. *Routinery application of the axiom (T).* □

Let G be the set of terms of CCCS over the restricted signature comprising nil, prefixing and summation only (thus, also a subset of terms of the equational theory $\mathcal{F}$). In other words, terms in G are given by the following syntax

$$G ::= nil\ ,\ t.G\ ,\ G + G$$

where t is a transition of Closed CCS. We have the following result.

Theorem 12 *(The Generalized Expansion Theorem for CCCS)*
For every term F of the equational theory $\mathcal{F}$ there exists a term G such that

$$\mathcal{F} \vdash F = G$$

Proof. *The proof is by induction on the structure of terms. The unique non trivial step concerns parallel terms of the form $F_1 \mid F_2$. In this case the theorem is proved by using the axioms EX, D1-D7 and Z, in order to push parallel composition inside the structure of the term and to remove it at the end.*□

Since CCCS agents are terms of the equational theory $\mathcal{F}$, we can safely say that the theorem above ensures the existence of *normal forms* for agents within the syntax of agents itself. The following theorem is the crux of the paper and shows that the proposed axiomatization is sound and complete (the proof follows [HM85]).

Theorem 13 *(Soundness and Completeness)*
$E_1 \sim E_2$ if and only if $\mathcal{F} \vdash E_1 = E_2$.
Proof.
All the axioms in $\mathcal{F}$ are satisfied[2] *by $\sim$.*
The converse, i.e. if $E \sim E'$ then $\mathcal{F} \vdash E = E'$, will be firstly proved for normal forms E and E', by inducing on their structure, noting that $E \xrightarrow{\omega} \overline{E}$ implies that $\overline{E}$ is a subterm of E.
Let $\equiv$ be the congruence on terms induced by the axioms of $\mathcal{F}$. We therefore assume that E and E' take the forms

$$E = \sum_m t_i.E_i$$

and

$$E' = \sum_n t^j.E^j$$

where transitions t_i and t^j have observations ω_i and ω^j, respectively. Assume that $E \sim E'$. Since $E \xrightarrow{\omega_i} E_i$, then for some $\underline{E}$, $E' \xrightarrow{\omega_i} \underline{E}$ and $E_i \sim \underline{E}$. But $\underline{E}$ must be E^j for some j, with $\omega_i = \omega^j$, and by induction $E_i \equiv E^j$. So such a j must exists for each i, and by symmetry for each i there exists j such that $E_i \equiv E^j$ also. It then follows from axioms S1-S4 and T that $E \equiv E'$.
Finally, when E and E' are arbitrary CCCS agents, Theorem 12 ensures that any agent can be proved congruent, and thus (by the first part of this theorem) observationally equivalent, to its normal form. □

[2] To be rigorous, we could not say that $\sim$ satisfies the axioms EX and D1-D7 because they involve also terms which are not CCCS agents. However, it is immediate to see that $\sim$ can be safely extended also to those terms.

4 An Alternative Formulation of CCCS

One of the main consequences of the equational characterization of the equivalence $\sim$ is that we can safely replace the algebra of transitions with an algebra of observations. In fact according to axiom T all transitions with the same observation have the same meaning within the prefix operator. Thus, in $t.E$ we can safely replace t with its observation ω. In this section, we introduce an equivalent but formally simpler definition of CCCS.

Definition 14 *(Algebra of Observations)*
An observation algebra Ω, *with elements called observations and denoted by* ω, *is any one-sorted algebra with the following syntax:*

$$\omega ::= \mu \,,\, \omega\backslash\alpha \,,\, \omega[\Phi] \,,\, \omega\rfloor - \,,\, -\lfloor\omega \,,\, \omega \mid \omega'$$

and satisfying the following axioms:

$$\delta\backslash\alpha = \delta[\Phi] = \delta\rfloor - = -\lfloor\delta = \omega \mid \delta = \delta \mid \omega = \delta.$$

□

For instance, $\alpha \mid \overline{\alpha}$, $(\alpha\rfloor-)$ and $(-\lfloor\beta)$ are observations. Indeed, it is easy to see that the labels of proper transitions (see Definition 3) plus δ, are the elements of the initial algebra in the class of algebras satisfying the above presentation. Notice that all the error transitions are now labelled with δ because the operations of any observation algebra are *strict* on the action δ.

Definition 15 *(Closed CCS: Parameterized Form)*
Let Ω be an observation algebra. A $CCCS_\Omega$ agent E has the following syntax:

$$E ::= nil \,,\, \omega.E \,,\, E\backslash\alpha \,,\, E[\Phi] \,,\, E_1 + E_2 \,,\, E_1 \mid E_2$$

where $\omega \in \Omega$ is an observation.
The operational semantics is defined by axioms and inference rules which allow us to derive statements of the form $E_1 \xrightarrow{\omega} E_2$.

$$\omega.E \xrightarrow{\omega} E$$

$$\frac{E_1 \xrightarrow{\omega} E_2}{E_1\backslash\alpha \xrightarrow{\omega\backslash\alpha} E_2\backslash\alpha}$$

$$\frac{E_1 \xrightarrow{\omega} E_2}{E_1[\Phi] \xrightarrow{\omega[\Phi]} E_2[\Phi]}$$

$$\frac{E_1 \xrightarrow{\omega} E_2}{E_1 + E \xrightarrow{\omega} E_2} \qquad \frac{E_1 \xrightarrow{\omega} E_2}{E + E_1 \xrightarrow{\omega} E_2}$$

$$\frac{E_1 \xrightarrow{\omega} E_2}{E_1 \mid E \xrightarrow{\omega\rfloor-} E_2 \mid E} \qquad \frac{E_1 \xrightarrow{\omega} E_2}{E \mid E_1 \xrightarrow{-\lfloor\omega} E \mid E_2}$$

$$\frac{E_1 \xrightarrow{\omega} E_2,\ E_1' \xrightarrow{\omega'} E_2'}{E_1 \mid E_1' \xrightarrow{\omega\mid\omega'} E_2 \mid E_2'}$$

□

Similar results on the axiomatization of the bisimulation equivalence can be easily proved also for this alternative formulation.

Definition 16 *(Parameterized Bisimulation)*
Let Ω be an observation algebra. A binary symmetric relation R on $CCCS_\Omega$ agents is called Ω-bisimulation *if and only if R satisfies the following clause:*

- *if $u_1 \, R \, u_2$, and $u_1 \xrightarrow{\omega_1} v_1$, $\omega_1 \neq \delta$, then there exists v_2 such that $u_2 \xrightarrow{\omega_2} v_2$, $v_1 \, R \, v_2$, and $\omega_1 = \omega_2$ when evaluated in Ω.*

Two Closed CCS terms, u_1 and u_2, are bisimilar, and we write $u_1 \sim_\Omega u_2$, if there exists a Ω-bisimulation relating them.□

As usual, relation $\sim_\Omega$ is the maximal Ω- bisimulation and it is an equivalence relation.

Definition 17 *(The Equational Theory $\mathcal{F}'$)*
The equational theory $\mathcal{F}'$ is the theory obtained from $\mathcal{F}$ by removing axiom T, by substituting everywhere observation ω for transition t, operations $\omega\rfloor-$, and $-\lfloor\omega$ for $t\rfloor E$, and $E\lfloor t$, and by replacing axiom N with the axiom $\delta.F = nil$.

$S1$ $\quad F + F = F$

$S2$ $\quad F_1 + F_2 = F_2 + F_1$

$S3$ $\quad F_1 + (F_2 + F_3) = (F_1 + F_2) + F_3$

$S4$ $\quad F + nil = F$

N $\quad \delta.F = nil,$

EX $\quad F_1 \mid F_2 = F_1\rfloor F_2 + F_1\lfloor F_2 + F_1 \parallel F_2$

$D1$ $\quad \omega.F_1\rfloor F_2 = (\omega\rfloor-).F_1 \mid F_2$

$D2$ $\quad F_2\lfloor\omega.F_1 = (-\lfloor\omega).F_2 \mid F_1$

$D3$ $\quad \omega_1.F_1 \parallel \omega_2.F_2 = (\omega_1 \mid \omega_2).F_1 \mid F_2$

$D4$ $\quad (F_1 + F_2)\rfloor F = F_1\rfloor F + F_2\rfloor F$

$D5$ $\quad F\lfloor(F_1 + F_2) = F\lfloor F_1 + F\lfloor F_2$

$D6$ $\quad (F_1 + F_2) \parallel F = F_1 \parallel F + F_2 \parallel F$

$D7$ $\quad F \parallel (F_1 + F_2) = F \parallel F_1 + F \parallel F_2$

$R1$ $\quad \omega.F\backslash\alpha = \omega\backslash\alpha.F\backslash\alpha$

$R2$ $\quad \omega.F[\Phi] = \omega[\Phi].F[\Phi]$

$R3$ $\quad (F_1 + F_2)\backslash\alpha = F_1\backslash\alpha + F_2\backslash\alpha$

$R4$ $\quad (F_1 + F_2)[\Phi] = F_1[\Phi] + F_2[\Phi]$

Z $\quad nil\backslash\alpha = nil[\Phi] = nil \mid nil = nil\rfloor F = F\lfloor nil = F \parallel nil = nil \parallel F = nil$

□

Theorem 18 *(Soundeness and Completeness)*
Let $\mathcal{A}_\Omega$ be an axiomatization of the observation algebra Ω. Then $E_1 \sim_\Omega E_2$ if and only if $\mathcal{F}' \cup \mathcal{A}_\Omega \vdash E_1 = E_2$.
Proof. *The axiomatization $\mathcal{A}_\Omega$ of the observation algebra is irrelevant to define normal forms and to establish their existence.* □

In this formulation of CCCS we can also introduce a programming logic in the style of *Hennessy – MilnerLogic* [HM85], whose modalities are parameterized with respect to the observation algebra Ω.

Definition 19 *(The Programming Logic HML(Ω))*
The syntax of the HML(Ω) is

$$\varphi ::= true \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid [\omega]\varphi.$$

The satisfaction relation $\models$ for the logic on the set CCCS agents is as follows:

- $E \models true$ *for any* E,
- $E \models \neg\varphi$ *if and only if* $E \not\models \varphi$
- $E \models \varphi_1 \vee \varphi_2$ *if and only if* $E \models \varphi_1$ *or* $E \models \varphi_2$
- $E \models [\omega]\varphi$ *if and only if for any* E' *such that* $E \xrightarrow{\omega} E'$, *we have* $E' \models \varphi$

□

Such family of programming logics naturally induces an equivalence relation $\equiv_\Omega$ on CCCS agents. We say that $E_1 \equiv_\Omega E_2$ if and only if $E_1 \models \varphi$ iff $E_2 \models \varphi$ for all formulae φ of the logic.

Theorem 20 *(Logical Characterization)*

$$E_1 \equiv_\Omega E_2 \Longleftrightarrow E_1 \sim_\Omega E_2$$

□

The proof of the theorem follows the same pattern of the proof given by Hennessy and Milner in [HM85][3].

5 Interpreting the Operations

As already anticipated, in this section we show that by suitably interpreting the operations on the algebra of observations it is possible to derive the classical interleaving expansion theorem and also a truly concurrent interpretation which faithfully handles partial ordering observations.

5.1 Recovering Interleaving Semantics

We now introduce the *Algebra $\mathcal{I}$* of *Interleaving Observations*. Its elements are the actions, including τ and δ. The interpretation of the operations is defined by the following axiomatization $\mathcal{A}_\mathcal{I}$:

$$\mu\backslash\alpha = \; if\; \mu \in \{\alpha, \overline{\alpha}\}\; then\; \delta,\; else\; \mu$$

$$\mu[\Phi] = \Phi(\mu)$$

$$\mu\rfloor - = \mu = -\lfloor\mu$$

$$\mu_1 \mid \mu_2 = \; if\; \mu_1 = \overline{\mu_2}\; then\; \tau,\; else\; \delta.$$

Clearly, the classical Interleaving Expansion Theorem is derivable in $\mathcal{F}' \cup \mathcal{A}_\mathcal{I}$. Indeed, in the equational system $\mathcal{F}' \cup \mathcal{A}_\mathcal{I}$, the following equalities hold:

[3] Notice that because we are dealing with finite CCCS agents we do not need to introduce the image finiteness requirement on the transition relation.

$$\mu.E_1 \rfloor E_2 = (\mu\rfloor -).(E_1 \mid E_2) = \mu.(E_1 \mid E_2)$$

$$E_1 \lfloor \mu.E_2 = (-\lfloor \mu).(E_1 \mid E_2) = \mu.(E_1 \mid E_2)$$

$$\alpha.E_1 \parallel \overline{\alpha}.E_2 = (\alpha \mid \overline{\alpha}).(E_1 \mid E_2) = \tau.(E_1 \mid E_2)$$

$$\alpha.E_1 \parallel \beta.E_2 = (\alpha \mid \beta).(E_1 \mid E_2) = \delta.(E_1 \mid E_2) \ \mathit{where}\ \alpha \neq \beta$$

It can be proved (see, e.g., [Gor91]) that the classical interleaving expansion theorem can be obtained by using the axioms stated above. In particular, if one is interested in finding the most compact representation in ACP-style of Milner's expansion theorem, note that parallel composition | becomes commutative, and thus a single auxiliary operator for asynchrony is enough (the topological information on observations is irremediably lost).
We have proved (see example 10) that in $\mathcal{F}$ we have:

$$\alpha.nil \mid \beta.nil = (\alpha\rfloor\beta.nil).(nil\lfloor\beta).nil + (\alpha.nil\lfloor\beta).(\alpha\rfloor nil).nil + (\alpha \mid \beta).nil$$

In $\mathcal{F}'$ we have:

$$\alpha.nil \mid \beta.nil = (\alpha\rfloor -).(-\lfloor\beta).nil + (-\lfloor\beta).(\alpha\rfloor -).nil + (\alpha \mid \beta).nil$$

Finally, in $\mathcal{F}' \cup \mathcal{A}_\mathcal{I}$ we have:

$$\alpha.nil \mid \beta.nil = \alpha.\beta.nil + \beta.\alpha.nil + \delta.nil$$

$$= \alpha.\beta.nil + \beta.\alpha.nil \ + nil$$

$$= \alpha.\beta.nil + \beta.\alpha.nil$$

Similarly, it is easy to see that

$$\alpha.nil \mid \overline{\alpha}.nil = \alpha.\overline{\alpha}.nil + \overline{\alpha}.\alpha.nil + \tau.nil$$

5.2 A Truly Concurrent Interpretation

Now we introduce an algebra $\mathcal{SH}$ of concrete partial ordering observations, where partial orderings are enriched with attaching points, like *Concurrent Histories* [DM87,FMM91a]. The observation ω out of a transition $E \xrightarrow{\omega} E'$ will be interpreted as a particular kind of concurrent history, called *spatial history*, where the process labels are structured with topological information. We remark that this is just one of the possible truly concurrent interpretations. Indeed, by taking (slight variants of) Petri nonsequential processes [GR83,DMM89] or Causal Streams [FMM91b] as carrier sets, we can get an observation algebra as well.

A *concurrent history* h is a triple $(V, \leq, \ell)$, where $(V, \leq)$ is a *partial ordering*, $\ell : V \rightarrow P \cup A$ is the *labelling function* which sends the set of maximal and minimal elements (w.r.t. $\leq$) to P (the set of *process types*) and all the other elements to A (the set of *event types*). The elements of h with labels in P are called *processes*, those with labels in A are called *events*. Minimal processes, denoted by $\mathcal{S}(h)$, are called *sources*, while maximal processes, denoted by $\mathcal{D}(h)$, are called *destinations*. Furthermore, we require that ℓ be injective on sources (destinations), namely no two minimal (maximal) processes have the same label. Therefore, we can safely identify sources (destinations) with the set of their labels.

Concurrent Histories are naturally equipped with an operation of concatenation. Given two histories $h_1 = (V_1, \leq_1, \ell_1)$ and $h_2 = (V_2, \leq_2, \ell_2)$, the concatenation $h_1 \,;\, h_2$ is defined provided that $\mathcal{S}(h_2) = \mathcal{D}(h_1)$. In such a case, the resulting history $h_1 \,;\, h_2$ is obtained by matching $\mathcal{S}(h_2)$ with $\mathcal{D}(h_1)$, i.e. by identifying the corresponding processes (of course, two processes match provided that they have the same label). The relation $\leq_1 \cup \leq_2$ is made transitively closed, and the matched processes, whenever they are neither minimal nor maximal elements in the resulting partial ordering, are erased. Figure 1 shows an example of the operation of concatenation between histories. Graphically, processes (events) are represented by circles (boxes), and the partial orderings are represented by their Hasse diagrams growing downwards.

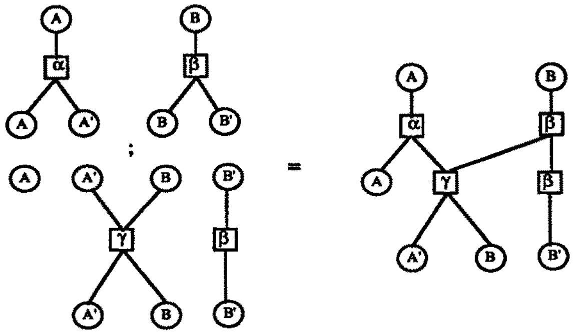

Figure 1: Two concurrent histories and their sequential composition

In order to introduce *Spatial Histories*, we have to define the set of process labels P. We remark that Spatial Histories are Concurrent Histories with certain structured process labels, called *localities*. Localities are given by:

$$L ::= \bullet\ ,\ L\rfloor -\ ,\ -\lfloor L$$

A set of localities is called a *chart*, and it is denoted by I, provided that it can be generated by the following rule:

$$\phi\ ,\ \{\bullet\}\ ,\ I\rfloor - \cup - \lfloor I \ \text{ are charts}$$

where ϕ is the empty chart and $I\rfloor - = \{p\rfloor - : p \in I\}$, i.e., the operator componentwise affects the elements of I (symmetrically for $-\lfloor I$). Finally, a chart is called *complete*, and is denoted by C, if it is generated by the following rule:

$$\{\bullet\}\ ,\ C\rfloor - \cup - \lfloor C \ \text{ are complete}$$

Notice that complete charts are charts, and that charts can be completed. Intuitively, charts describe information about the physical distribution of processes. A locality is very close to the notion of *grape*, introduced in [DM87,DDM88a,DDM90a], to deal with distributed semantics for CCS and related languages. A complete chart C gives a syntactical representation of a distributed system which occupies all the available space with its components. A chart I gives a consistent, representation of the distributed structure of a system which possibly leaves some space empty.

On charts a partial ordering relation $\preceq$ is easily defined: it is the least relation satisfying the following clauses:

(i) $\phi \preceq I$,

(ii) $\{\bullet\} \preceq C$,

(iii) $\dfrac{I_1 \prec I_2 \ and\ I_3 \prec I_4}{(I_1\rfloor - \cup - \lfloor I_3) \preceq (I_2\rfloor - \cup - \lfloor I_4)}$.

It is immediate to show that relation $\preceq$ is a partial ordering. Moreover, the set of charts with relation $\preceq$ is a *join semilattice*. This means that, given two charts I_1 and I_2, there always exists their greatest lower bound, $I_1 \sqcup I_2$, which is called the *most general unifier* (mgu) of I_1 and I_2. Procedurally, the chart $I_1 \sqcup I_2$ is built by adding them the minimal topological detail which makes the two charts in full agreement. For instance, let us consider the two charts $I_1 = \{\bullet\rfloor -\}$ and $I_2 = \{(\bullet\rfloor -)\rfloor -\}$. The chart $I_1 \sqcup I_2$ turns out to be $\{(\bullet\rfloor -)\rfloor -, (-\lfloor \bullet)\rfloor -\}$. If in proving that $I_1 \preceq I_1 \sqcup I_2$, and $I_2 \preceq I_1 \sqcup I_2$ we do not use the full power of axiom (i) but just the fact that $\phi \preceq \phi$, then I_1 and I_2 are called *strongly* unifiable. Intuitively, two strongly unifiable charts either are both complete, or leave the same empty space.

A *spatial history* is a concurrent history $h = (V, \leq, \ell)$, where process types are localities (the labelling function ℓ sends sources and destinations to charts) and event types are basic actions; moreover we require that sources and destinations are strongly unifiable, and that no process is both a minimal and a maximal element of the partial ordering. Finally, a spatial history has no events labelled by δ. Spatial Histories are considered up to isomorphisms of labelled partial orders.

We now introduce an observation algebra whose elements are spatial histories plus the error δ. To simplify the definition of the algebra, we first introduce some notations. We write h_μ for the history with three linearly ordered elements where the middle one is labelled by μ, and the two extrema by $\bullet$. Clearly, the constants $\Lambda \cup \{\tau\}$ will be interpreted as the histories h_μ, and the constant δ by the element δ of the algebra. As required by Definition 14 δ is absorbent with respect to all the operations.

The interpretation of the operations is given as follows. We have $h = (V, \leq, \ell)$.

- $h\rfloor- = (V, \leq, \ell')$, where, for all processes v, $\ell'(v) = \ell(v)\rfloor-$, and for all the events v, $\ell'(v) = \ell(v)$. It is immediate to observe that if h satisfies the requirements about strong unifiability of sources with destinations, then also $h\rfloor-$ does, and thus also $h\rfloor-$ is a spatial history. Symmetrically for the operation $-\lfloor h$.
- The operation of synchronization, $h_1 \mid h_2$, yields a history only when both h_1 and h_2 have a unique event - say, v_1 and v_2, respectively - and $\ell_1(v_1) = \overline{\ell_2(v_2)}$. Otherwise, $h_1 \mid h_2 = \delta$. In the first case, the spatial history $h_1 \mid h_2 = (V, \leq, \ell)$ is given as follows. Set V is equal to $(V_1 \uplus V_2)$, where $\uplus$ is the union of V_1 and V_2, obtained by *identifying* the elements v_1 and v_2, (the resulting element is denoted by $\overline{v}$). Relation $\leq$ is the transitive closure of the union of the two relations $\leq_1$ and $\leq_2$ on V. The labelling function ℓ is:
  - $\ell(\overline{v}) = \tau$;
  - for all $v \in V_1, \ell(v) = \ell_1(v)\rfloor-$;
  - for all $v \in V_2$, $\ell(v) = -\lfloor \ell_2(v)$.
- $h\backslash\alpha$ yields a history only when there are no events labelled by α or by $\overline{\alpha}$, and the result of the operation is h itself. Otherwise, $h\backslash\alpha$ yields δ.
- $h[\Phi]$ is the history obtained by applying the relabelling function Φ to events.

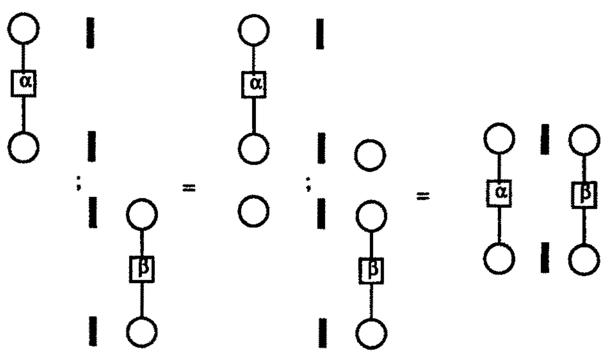

Figure 2: The procedure giving the concatenation $\alpha\rfloor-; -\lfloor\beta$ of spatial histories $\alpha\rfloor-$ and $-\lfloor\beta$

Now it could be interesting to concatenate histories in order to have a complete observation out of a computation. To this aim, we introduce a concatenation operation on spatial histories for growing the partial orderings.

Given two spatial histories h_1 and h_2, the concatenation $h_1; h_2$ is the spatial history obtained by firstly substituting $\mathcal{D}(h_1) \sqcup \mathcal{S}(h_2)$ for $\mathcal{D}(h_1)$ in h_1 and for $\mathcal{S}(h_2)$ in h_2, and then concatenating h_1 and h_2 according to the concatenation operation given above for concurrent histories. Notice that this operation is well defined because $\mathcal{D}(h_1) \sqcup \mathcal{S}(h_2)$ always exists. An example of this construction is reported in Figure 2. The localities of processes are directly drawn in the picture. For instance, the label of the upmost leftmost process in the picture is $\bullet\rfloor -$, as it can be inferred by its position with respect to the thick symmetry line.

As an example of the expansion theorem in the case of the algebra of truly concurrent observations given by spatial histories, let us consider the CCCS agent $(\alpha.(\beta \mid nil) \mid \overline{\beta}.nil)\backslash\beta$. According to the expansion theorem and to the interpretation of the operations we have:

$$(\alpha.(\beta \mid nil) \mid \overline{\beta}.nil)\backslash\beta = (\alpha\rfloor -).((\beta\rfloor -) \mid \overline{\beta}).nil$$

The spatial history observation of the unique 2-step computation is illustrated in Figure 3.

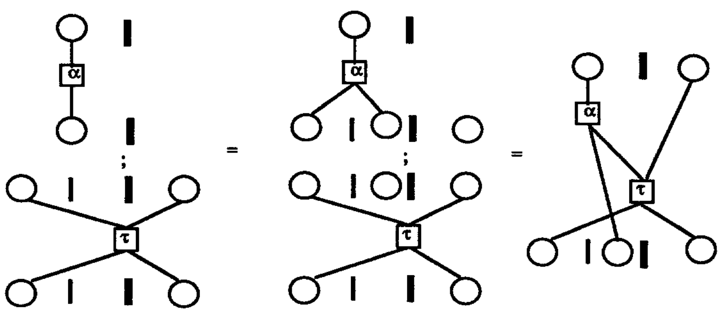

Figure 3: The observation of the 2-step computation of $(\alpha.(\beta \mid nil) \mid \overline{\beta}.nil)\backslash\beta$

We are presently investigating an axiomatization of the algebra $\mathcal{SH}$ of spatial histories, following the intuition behind the proposal presented in [FM90]. There, a truly concurrent axiomatization of CCS is given by exploiting a relation stating that independent transitions can be permuted. We plan to apply a similar machinery to spatial histories; indeed, spatial histories involving different localities are obviously independent and thus can be permuted.

6 Extensions and Future Work

The first natural extension concerns the introduction of a recursive definition construct, like for CCS. To this aim, we can extend the syntax in the obvious way, and adapt the overall theory without too much effort. Of course, no finite axiomatization is given in the general case, except for regular behaviours [Mil89].

As soon as we admit transitions as arguments of the prefixing operator, we augment the expressive power of the language, not only because the axiomatization we propose would not be possible otherwise, but also because the algebra of transitions (and thus of observations) can be enriched as one wishes. For example, here we discuss the relevant case in which sequential composition, denoted by ";", is added to the transition algebra. In this way, atomically executed computations can be arguments for prefixing, thus introducing explicitly a mechanism to handle *atomic actions*. In the context of Process Algebras, the notion of atomic action has been extensively used to address the problem of action refinement. Some preliminary results are reported in ([BC88], [dBK88], [GMM90,DG90,Gor91]).

In order to introduce the operation of sequential composition and the mechanism of atomicity, we need two transition relations, the former ($\longrightarrow$) for transitions, the latter ($\Longrightarrow$) for computations. Of course, any transition is a computation and thus we need the following rule:

$$\frac{t : E_1 \xrightarrow{\omega} E_2}{t : E_1 \stackrel{\omega}{\Longrightarrow} E_2}$$

Moreover, we need the rule for concatenating computations, which induces also a corresponding operation on observations:

$$\frac{t : E_1 \stackrel{\omega}{\Longrightarrow} E_2, t' : E_2 \stackrel{\omega'}{\Longrightarrow} E_3}{t; t' : E_1 \stackrel{\omega;\omega'}{\Longrightarrow} E_3}$$

Finally, we have to transform a (possibly multi-step) computation in just an atomic transition. To this aim, the inference rule for prefixing is replaced by the following:

$$\frac{t : E_1 \stackrel{\omega}{\Longrightarrow} E_2}{[t, E >: t.E \xrightarrow{\omega} E}$$

Bisimulation is defined considering only the atomic relation $\rightarrow$, and all the results proved in the paper hold also in this more general case.

With the introduction of these new rules, the language becomes extensively strenghtened. However, a difficult problem arises, namely the relationship between the operations of sequential composition and synchronization. Indeed, the synchronization operation does not have a natural counterpart on computations. For instance, in an interleaving setting, [GMM90] introduces an operation on sequences of actions which is intrinsically nondeterministic, and thus is not expressible in our algebraic framework. An alternative approach has been followed in a truly concurrent setting by [GM90], where a restricted (partial) synchronization operation on processes has been proposed.

References

[AB84] Austry, D. Boudol, G. *Algebre de Processus et Synchronization*, Theoretical Computer Science **30** (1), pp 91-131, 1984.

[BC88] Boudol, G., Castellani, I., *Concurrency and Atomicity,* Theoretical Computer Science **59** (1,2), pp. 25- 84, 1988.

[BC89] Boudol, G., Castellani, I., *Permutations of Transitions: An Event Structure Semantics for CCS,* In Proc. REX School, Workshop on Linear Time Branching Time and Partial Orders in Logics and Models for Concurrency, LNCS **354**, pp 411-437 1989.

[BC90] Boudol, G., Castellani, I., *Three Equivalent Semantics Semantics for CCS,* In Semantics of Systems of Concurrent Processes, LNCS **469**, pp 96-141, 1990.

[dBK88] de Bakker, J.W., Kok, J.N., *Uniform Abstraction, Atomicity and Contractions in the Comparative Semantics of Concurrent Prolog,* in Proc. of Int. Conf. on Fifth Generation Computer Systems Tokyo, 1988.

[BHR84] Brookes, S.D., Hoare, C.A.R., Roscoe, A.D., *A Theory of Communicating Sequential Processes,* Journal of ACM **31** (3), pp 560-599, 1984.

[BK84] Bergstra, J., Klop, W., *Process Algebra for Synchronous Communication,* Information and Control **60**, pp 109-137, 1984.

[BK85] Bergstra, J., Klop, W., *Algebra of Communicating Processes with Abstraction,* Theoretical Computer Science, **37**, (1), pp 77-121, 1985.

[DD89] Darondeau, P., Degano, P. *Causal Trees,* In Proc. ICALP 89, LNCS **372**, pp 234-248, 1989.

[DD90] Darondeau, P., Degano, P., *Causal Tree = Interleaving + Causality,* In Semantics of Systems of Concurrent Processes, LNCS **469**, pp 239-255, 1990.

[DDM85] Degano, P., De Nicola, R., Montanari, U. *Partial ordering derivations for CCS,* Proc. 5th Int. Conf. Fundamentals of Computation Theory, LNCS **199**, pp. 520-533, 1985.

[DDM87] Degano, P. De Nicola, R., Montanari, U. *Observational Equivalences for Concurrency Models*, in Formal Description of Programming Concepts III, M. Wirsing, ed., North Holland, pp 105-132, 1987.

[DDM88a] Degano, P. De Nicola, R., Montanari, U. *A Distributed Operational Semantics for CCS Based on Condition/Event Systems.* Acta Informatica, **26**, pp 59-91, 1988.

[DDM88b] Degano, P. De Nicola, R., Montanari, U. *On the Consistency of Truly Concurrent Operational and Denotational Semantics.* Proc. LICS 88, IEEE Computer Society Press, pp 133-141, 1988.

[DDM89] Degano, P. De Nicola, R., Montanari, U. *Partial Ordering Descriptions and Observations of Nondeterministic Concurrent Processes*, in Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, LNCS **354**, pp 438- 466, 1989.

[DDM90a] Degano, P. De Nicola, R., Montanari, U. *A Partial Ordering Semantics for CCS*, Theoretical Computer Science (75), pp. 223-262, 1990.

[DDM90b] Degano, P., De Nicola, R., Montanari, U. *On the Operational Semantics of Distributed Concurrent Systems* in: Declarative Systems (G. David, R.T. Boute, and B.D. Shriver Eds) Proc. IFIP 10.1 Workshop on Concepts and Characteristics of Declarative Systems, North-Holland, Amsterdam, 1990.

[DDM91] Degano, P. De Nicola, R., Montanari, U. *Universal Axioms for Bisimulation*, Submitted for Publication, 1991.

[DG90] Degano, P., Gorrieri, R., *Atomic Refinement in Process Description Languages*, Submitted for Publication, 1990.

[DMM89] Degano, P., Meseguer, J., Montanari, U., *Axiomatizing Net Computations and Processes*, in LICS 89, IEEE Computer Society Press, pp 175-185, 1989.

[DM87] Degano, P., Montanari, U., *Concurrent Histories: A Basis for Observing Distributed Systems*, Journal of Computer and System Sciences **34**, pp 422-461, 1987.

[Fe90] Ferrari, G. *Unifying Models of Concurrency*, PhD Thesis, TD/4- 90, Dipartimento di Informatica, Univ. Pisa, 1990.

[FM90] Ferrari, G., Montanari, U., *Towards the Unification of Models for Concurrency*, Proc. CAAP'90, LNCS **431**, pp 162-176, 1990.

[FMM91a] Ferrari, G., Montanari, U., Mowbray, M. *On Causality Observed Incrementally, Finally*, To appear Proc. TAPSOFT - CAAP 91, LNCS, 1991.

[FMM91b] Ferrari, G., Montanari, U., Mowbray, M. *Causal Streams: Tracing Causality in Distributed Systems*, Submitted for publication, 1991.

[G88] Gischer, J. *The Equational Theory of Pomsets*, Theoretical Computer Science **61**, 1988.

[Go88] Goltz, U. *On Representing CCS Programs by Finite Petri Nets*, Proc. MFCS 88, LNCS **324**, pp 339-350, 1988.

[Gor91] Gorrieri, R. *Refinement, Atomicity and Transactions for Process Description Languages*, PhD Thesis, Dipartimento di Informatica, Univ. Pisa, 1991.

[Gr81] Grabowski, J. *On Partial Languages*, Fundamenta Informaticae, IV 2, pp 427-498, 1981.

[GG89] van Glabbeek, R., Goltz, U., *Equivalence Notions for Concurrent Systems and Refinement of Actions*, Proc. MFCS '89, LNCS **397**, pp 237-248, 1989.

[GM90] Gorrieri, R., Montanari, U., *SCONE: a Simple Calculus of Nets*, In Proc. CONCUR'90, LNCS **458**, pp 2-30, 1990.

[GMM90] Gorrieri, R., Marchetti, S., Montanari, U. *A^2CCS:Atomic Actions for CCS*, Theoretical Computer Science, **72**, pp 203-223 1990.

[GR83] Goltz, U., Reisig, W., *The Non Sequential Behaviour of Petri Nets*, In formation and Computation **57**, pp 125-147, 1983.

[Ho85] Hoare, C.A.R., *Communicating Sequential Processes,* Prentice Hall, 1985.

[HM85] Hennessy, M., Milner, R., *Algebraic Laws for Nondeterminism and Concurrency,* Journal of ACM **32** (1), pp 137-141, 1985.

[He88] Hennessy, M. *An Algebraic Theory of Processes*, MIT Press, 1988.

[Mil80] Milner, R., *A Calculus of Communicating Systems,* LNCS **92**, 1980.

[Mil89] Milner, R., *Communication and Concurrency,* Prentice Hall, 1989.

[Mol90a] Moller, F. *The Nonexistence of Finite Axiomatizations for CCS Congruences*, Proc. LICS'90, IEEE Press, pp 142-153 1990.

[Mol90b] Moller, F. *The Importance of the Left Merge Operator in Process Algebras,* Proc. ICALP'90, LNCS **443**, pp 752-764, 1990.

[MM90] Meseguer,J. Montanari, U. *Petri Nets are Monoids: A new Algebraic Formulation of Net Theory,* Information and Computation, **88** (2), pp 105-155, 1990.

[MY89] Montanari, U. Yankelevich, D. An Algebraic View of Interleaving and Distributed Operational Semantics, In Proc. Third Symposium on Category Theory and Computer Science, LNCS **389**, pp 5-20, 1989.

[Par81] Park, D., *Concurrency and Automata on Infinite Sequences,* in Proc. GI, LNCS **104**, pp 167-183, 1981.

[NPW81] Nielsen, M., Plotkin, G. Winskel, G. *Petri Nets, Event Structures and Domains* (Part I). Theoretical Computer Science, **13** (1), pp 85-108, 1981.

[Ol87] Olderog, E.-R., *Operational Petri Nets Semantics for CCSP,* in Advances in Petri Nets 87, LNCS **266**, pp 196-223, 1987.

[Plo81] Plotkin, G. A Structured Approach to Operational Semantics, DAIMI FN-19, Computer Science Dept. University of Aarhus, 1981.

[Pr86] Pratt, V., *Modelling Concurrency with Partial Orders,* Int. Journal of Parallel Programming, **15** (1), pp 381-400, 1986.

[RT88] Rabinovich, A. Trakhtenbrot, B. *Behaviour Structures and Nets,* Fundamenta Informaticae XI, pp 357-404,1988.

[Re85] Reisig, W. *Petri Nets: An Introduction*, EATCS Monograph, Springer, 1985.