Observing distribution in processes

Ilaria Castellani*
INRIA Sophia-Antipolis
06565 Valbonne, France
email: ic@sophia.inria.fr

Abstract. The distributed structure of CCS processes can be made explicit by assigning different *locations* to their parallel components. These locations then become part of what is observed of a process. The assignment of locations may be done statically, or dynamically as the execution proceeds. The dynamic approach was developed first, by Boudol et al. in [BCHK91a], [BCHK91b], as it seemed more convenient for defining notions of *location equivalence* and *preorder*. However, it has the drawback of yielding infinite transition system representations. The static approach, which is more intuitive but technically more elaborate, was later developed by L. Aceto [Ace91] for *nets of automata*, a subset of CCS where parallelism is only allowed at the top level. In this approach each net of automata has a finite representation, and one may derive notions of equivalence and preorder which coincide with the dynamic ones. The present work generalizes the static treatment of Aceto to full CCS. The result is a distributed semantics which yields finite transition systems for all CCS processes with a regular behaviour and a finite degree of parallelism.

1 Introduction

This work is concerned with *distributed semantics* for CCS, accounting for the spatial distribution of processes. Such semantics focus on different aspects of behaviour than most non-interleaving semantics for CCS considered so far in the literature, which are based on the notion of causality. Roughly speaking, a distributed semantics keeps track of the behaviour of the local components of a system, and thus is appropriate for describing phenomena like a local deadlock. On the other hand a causal semantics, such as those described in [DDNM87], [GG89], [DD90] [BC91], is concerned with the flow of causality among activities and thus is better suited to model the interaction of processes and the global control structure of a system.

The distributed structure of CCS processes can be made explicit by assigning different *locations* to their parallel components. To this end we use the *location prefixing* construct $l :: p$ of [BCHK91a,b], which represents process p residing at location l. The actions of such a process are observed together with their location. We have for instance:

$$(l :: a \mid k :: b) \xrightarrow[l]{a} (l :: nil \mid k :: b) \xrightarrow[k]{b} (l :: nil \mid k :: nil)$$

In general, because of the nesting of parallelism, the locations of actions will not be simple letters $l, k, \ldots$ but rather words $u = l_1 \cdots l_n$. Then a "distributed process" will perform transitions of the form $p \xrightarrow[u]{a} p'$.

The idea is now to compare CCS terms by comparing their possible *distributions*, which are obtained by transforming each subprocess $(p \mid q)$ into $(l :: p \mid k :: q)$, where l and k are distinct locations. Intuitively, such an assignment of locations should be done statically, and

*This work has been partly supported by the Project 502-1 of the Indo-French Centre for the Promotion of Advanced Research.

then become part of what is observed of a process. This will allow us to distinguish for example $(a \mid b)$ from $(a.b + b.a)$, since any distribution of the first process will perform actions a and b at different locations. For more interesting examples we refer the reader to the introductions of [BCHK91a,b].

We thus want to define a notion of (weak) bisimulation on distributed processes, which equates processes exhibiting the same "location transitions". However, it would be too strong a requirement to ask for the identity of locations in corresponding transitions. In fact, if we want to observe distribution, we still aim, to some extent, at an extensional semantics. For instance, we do not want to observe the order in which parallel components have been assembled in a system, nor indeed the number of these components. We are only interested in the components which are active in each computation. We would like e.g. to identify the distributions of the CCS processes:

$$a \mid (b \mid c) \quad \text{and} \quad (a \mid b) \mid c$$
$$a \quad \text{and} \quad a \mid nil$$

Then transitions must be compared modulo an *association* between their locations. For instance to relate the distributions $l :: a \mid k :: (l' :: b \mid k' :: c)$ and $l :: (l' :: a \mid k' :: b) \mid k :: c$, we need to "identify" the locations l, kl', kk' of the first respectively with ll', lk', k in the second. However, it appears that this association cannot in general be fixed statically. For consider the two CCS processes:

$$p = [(\alpha + b) \mid \bar{\alpha}.b]\backslash\alpha \quad \text{and} \quad q = b$$

Intuitively, we would like to equate p and q because the observable behaviour of any distributions of these processes consists in just one action b at some location l. But here the required association of locations will depend on which run is chosen in the first process. Hence it is not obvious how to define a notion of equivalence formalising our intuition about abstract distributed behaviours.

Because of this difficulty, the static approach was initially abandoned in favour of a different one, where locations are introduced dynamically as the execution proceeds. This *dynamic approach*, where locations are associated with actions rather than with parallel components, has been presented in [BCHK91a,b]. In this setting, the notion of *location equivalence* is particularly simple: it is just the standard notion of bisimulation, applied to the transitions $p \xrightarrow[u]{a} p'$. Moreover, by weakening a little the definition of the equivalence, we obtain a notion of *location preorder*, which formalises the idea that one process is more sequential or less distributed than another. Such a notion is particularly useful when dealing with truly concurrent semantics, where an implementation is often not equivalent to its - generally more sequential - specification. Since location equivalence and preorder are essentially bisimulation relations, many proof techniques familiar from the theory of standard bisimulation may be applied to them: for example both these relations have a complete axiomatisation and a logical characterisation in the style of Hennessy and Milner, see [BCHK91a,b].

However, the dynamic approach has the drawback of yielding infinite transition systems even for regular processes, and thus cannot be directly used for verification purposes. Moreover in this approach locations represent access paths for actions rather than sites in a system, and thus are somehow remote from the original intuition. For these reasons, it was interesting to resume the initial attempt at a static approach. The problem of finding the appropriate notion of bisimulation was solved by L. Aceto in [Ace91] for *nets of automata*, a subset of CCS where parallelism is only allowed at the top level. The key idea here is to replace the usual notion of a bisimulation relation by that of a family of relations indexed by location associations. Aceto

shows that the notions of static location equivalence and preorder thus obtained coincide with the dynamic ones, and thus may be used as "effective" versions of the latter.

The purpose of the present work is to generalize the static treatment of Aceto to full CCS. Having established the notion of distribution for general CCS processes, the main point is to adapt Aceto's definitions of static location equivalence and preorder. Because of the arbitrary nesting of parallelism and prefixing in CCS terms, and of the interplay between sum and parallelism, this is not completely straightforward. A step in this direction was done recently by Mukund and Nielsen in [MN92], where a notion of bisimulation equivalence based on static locations is proposed for a class of asynchronous transition systems modelling CCS with guarded sums. The notion of equivalence we present here is essentially the same (extended to all CCS), and our main result is that it coincides with the dynamic location equivalence of [BCHK91b]. We also show that a similar result holds for the location preorders.

A transition system for CCS labelled with static locations, called "spatial transition system", has been also presented in [MY92], [Yan93]. Here locations are essentially used to build a second transition system, labelled by partial orders representing *local causality*, on which is based the theory of equivalence (as well as of preorders, in [Yan93]). Again, this partial order transition system gives finite representations only for finite behaviours. This work also confirms what had been previously shown by A.Kiehn in [Kie91], namely that observing dynamic locations amounts to observe *local causality* in computations. In [Kie91] one may also find a detailed comparison of distributed and causality-based semantics.

In this extended abstract all proofs are omitted. For a full account we refer to the complete version of the paper [Cas93].

2 A language for processes with locations

We introduce here a language for specifying processes with locations, called LCCS. This language is a simple extension of CCS, including a new construct to deal with locations.

We start by recalling some conventions of CCS [Mil80]. One assumes a set of names Λ, ranged over by $\alpha, \beta, \ldots$, and a corresponding set of co-names $\bar{\Lambda} = \{\bar{\alpha} \mid \alpha \in \Lambda\}$, where $\bar{\ }$ is a bijection such that $\bar{\bar{\alpha}} = \alpha$ for all $\alpha \in \Lambda$. The set of visible actions is given by $Act = \Lambda \cup \bar{\Lambda}$. Invisible actions – representing internal communications – are denoted by the symbol $\tau \notin Act$. The set of all actions is then $Act_\tau =_{\text{def}} Act \cup \{\tau\}$. We use $a, b, c, \ldots$ to range over Act and $\mu, \nu, \ldots$ to range over Act_τ. We also assume a set V of process variables, ranged over by $x, y \ldots$.

In addition to the operators of CCS, which we suppose the reader to be familiar with, LCCS includes a construct for building processes with explicit locations. Let *Loc*, ranged over by $l, k, \ldots$, be an infinite set of atomic locations. The new construct of *location prefixing*, noted $l :: p$, is used to represent process p residing at location l. Intuitively, the actions of such a process will be observed "within location l". The syntax of LCCS is as follows:

$$p ::= nil \mid \mu.p \mid (p \mid q) \mid (p + q) \mid p\backslash\alpha \mid p\langle f \rangle \mid x \mid rec\, x.\, p \mid l :: p$$

We use the slightly nonstandard notation $p\langle f \rangle$ to represent the relabelling operator of CCS. In a previous paper [BCHK91b], this language has been given a location semantics based on a dynamic assignment of locations to processes. Here we shall present a location semantics based on a static notion of location, and show that the two approaches, dynamic and static, give rise to the same notions of equivalence and preorder on CCS processes. The basic idea, common to both approaches, is that the actions of processes are observed together with the locations at which they occur. In general, because of the nesting of parallelism and prefixing in terms, the locations of actions will not be atomic locations of *Loc*, but rather *words* over these locations.

Thus general locations will be elements $u, v \ldots$ of Loc^*, and processes will be interpreted as performing transitions

$$p \xrightarrow[u]{\mu} p'$$

where μ is an action and u is the location at which it occurs.

However, locations do not have the same intuitive meaning in the two approaches. In the static approach locations represent sites - or parallel components - in a distributed system, much as one would expect. In the dynamic approach, on the other hand, the location of an action represents the sequence of actions which are locally necessary to enable it, and thus is more properly viewed as an access path to that action within the component where it occurs. Because of this difference in intuition, it is not immediately obvious that the two approaches should yield the same semantic notions. The fact that they do means that observing distribution is essentially the same as observing local causality.

3 Static approach

We start by presenting an operational semantics for LCCS based on the static notion of location. The idea of this semantics is very simple. Processes of LCCS have some components of the form $l :: p$, and the actions arising from these components are observed together with their location. The distribution of locations in a term remains fixed through execution. Location prefixing is a static construct and the operational rules do not create new locations; they simply exhibit the locations which are already present in terms. Formally, this is expressed by the operational rules for action prefixing and location prefixing. Recall that locations are words $u, v, \ldots \in Loc^*$. The empty word ε represents the location of the overall system. The rules for $\mu.p$ and $l :: p$ are respectively:

(S1) $\quad \mu.p \xrightarrow[\varepsilon]{\mu}_s p$

(S2) $\quad p \xrightarrow[u]{\mu}_s p' \quad \Rightarrow \quad l :: p \xrightarrow[lu]{\mu}_s l :: p'$

Rule (S1) says that an action which is not in the scope of any location l is observed as a global action of the system. Rule (S2) shows how locations are transferred from processes to actions. The rules for the remaining operators, apart from the communication rule, are similar to the standard interleaving rules for CCS, with transitions $\xrightarrow[u]{\mu}_s$ replacing the usual transitions $\xrightarrow{\mu}$. The set of all rules specifying the operational semantics of LCCS is given in Figure 1. The rule for communication (S4) requires some explanation. In the strong location transition system we take the location of a communication to be that of the smallest component which includes the two communicating subprocesses: the notation $u \sqcap v$ in rule (S4) stands for the longest common prefix of u and v. For instance we have:

Example 3.1

$$l :: \alpha \mid k :: \bar{\alpha}.(l' :: \beta \mid k' :: \bar{\beta}) \xrightarrow[\varepsilon]{\tau}_s l :: nil \mid k :: (l' :: \beta \mid k' :: \bar{\beta}) \xrightarrow[k]{\tau}_s l :: nil \mid k :: (l' :: nil \mid k' :: nil)$$

However, we shall mostly be interested here in the *weak location transition system*, where τ-transitions will have no explicit location: since the transitions themselves are not observable, it would not make much sense to attribute a location to them. The weak location transitions $\overset{a}{\underset{u}{\Longrightarrow}}_s$ and $\overset{\tau}{\Longrightarrow}_s$ are thus defined by:

$$p \overset{\tau}{\Longrightarrow}_s q \quad \Leftrightarrow \quad \exists\, u_1, \ldots, u_n, p_0, \ldots, p_n \quad s.t. \quad p = p_0 \xrightarrow[u_1]{\tau}_s p_1 \cdots \xrightarrow[u_n]{\tau}_s p_n = q$$

$$p \overset{a}{\underset{u}{\Longrightarrow}}_s q \quad \Leftrightarrow \quad \exists\, p_1, p_2 \quad s.t. \quad p \overset{\tau}{\Longrightarrow}_s p_1 \xrightarrow[u]{a}_s p_2 \overset{\tau}{\Longrightarrow}_s q$$

We shall use the weak location transition system as the basis for defining a new semantic theory for CCS, and in particular notions of equivalence and preorder which account for the degree of distribution of processes. The reader may have noticed, however, that applying the rules of Figure 1 to CCS terms just yields a transition $p \xrightarrow[\varepsilon]{\mu}_s p'$ whenever the standard semantics yields a transition $p \xrightarrow{\mu} p'$. In fact, we shall not apply these rules directly to CCS terms. Instead, the idea is to first bring out the parallel structure of CCS terms by assigning locations to their parallel components, thus transforming them into particular LCCS terms which we call "distributed processes", and then execute these according to the given operational rules. The set DIS $\subseteq$ LCCS of *distributed processes* is given by the grammar:

$$p ::= nil \;\mid\; \mu.p \;\mid\; \underbrace{(l::p \mid k::q)}_{l \neq k} \;\mid\; (p+q) \;\mid\; p\backslash\alpha \;\mid\; p\langle f\rangle \;\mid\; x \;\mid\; rec\,x.\,p$$

Essentially, a distributed process is obtained by inserting a pair of distinct locations in a CCS term wherever there occurs a parallel operator. This is formalised by the notion of *distribution*.

Definition 3.2 The *distribution relation* is the least relation $\mathcal{D} \subseteq (\text{CCS} \times \text{DIS})$ satisfying:

- $nil\;\mathcal{D}\;nil$ and $x\,\mathcal{D}\,x$
- $p\,\mathcal{D}\,r \;\Rightarrow\; \mu.p\;\mathcal{D}\;\mu.r$
  $p\backslash\alpha\;\mathcal{D}\;r\backslash\alpha$
  $p\langle f\rangle\;\mathcal{D}\;r\langle f\rangle$
  $(rec\,x.\,p)\;\mathcal{D}\;(rec\,x.\,r)$
- $p\,\mathcal{D}\,r \;\&\; q\,\mathcal{D}\,s \;\Rightarrow\; (p \mid q)\;\mathcal{D}\;(l::r \mid k::s),\;\; \forall l,k \text{ s.t. } l \neq k$
  $(p+q)\;\mathcal{D}\;(r+s)$

If $p\,\mathcal{D}\,r$ we say that r is a *distribution* of p.

Note that the same pair of locations may be used more than once in a distribution. We shall see in fact, at the end of this section, that distributions involving just *two* atomic locations are sufficient for describing the distributed behaviour of CCS processes.

3.1 Static location equivalence

We want to define an equivalence relation $\approx_\ell^s$ on CCS processes, based on a bisimulation-like relation between their distributions. The intuition for two CCS processes p, q to be equivalent is that there exist two distributions of them, say $\bar{p}$ and $\bar{q}$, which perform "the same" location transitions at each step. However, as we argued already in the introduction, we cannot require the identity of locations in corresponding transitions. If we want to identify the following processes:

$$a \mid (b \mid c) \quad \text{and} \quad (a \mid b) \mid c$$
$$a \quad \text{and} \quad a \mid nil$$

it is clear that, whatever distributions we choose, we must allow corresponding transitions to have different – although somehow related – static locations. In general transitions will be compared modulo an *association* between their locations. The idea is directly inspired from that used by Aceto for nets of automata [Ace91]; however in our case the association will not be a bijection as in [Ace91], nor even a function. For example, in order to equate the two processes:

$$a.(b.c \mid nil) \quad \text{and} \quad a.b.(c \mid nil)$$

we need an association containing the three pairs $(\varepsilon,\varepsilon),(l,\varepsilon),(l,l')$, for some $l,l' \in Loc$.

In fact, the only property we will require of location associations is that they respect independence of locations. To make this precise, let $\ll$ denote the prefix ordering on Loc^*. If $u \ll v$ we say that v is an extension or a *sublocation* of u. If $u \not\ll v$ and $v \not\ll u$, what we indicate by $u \diamond v$, we say that u and v are *independent*.

Definition 3.3 A relation $\varphi \subseteq (Loc^* \times Loc^*)$ is a *consistent location association (cla)* if:

$$(u,v) \in \varphi \;\;\&\;\; (u',v') \in \varphi \;\;\Rightarrow\;\; (u \diamond u' \;\Leftrightarrow\; v \diamond v')$$

Essentially the same notion of consistent association has been proposed in [MN92] for a class of asynchronous transition systems modelling CCS with guarded sums.

Now Aceto showed in [Ace91] that, for a given pair of distributed processes we want to equate, the required *cla* cannot in general be fixed statically, but has to be built incrementally. For consider the two distributed processes, which are intuitively equivalent since both perform actions a and b in either order at different locations:

$$(\,l::(a.\gamma + b.\bar{\gamma}) \mid k::(\bar{\gamma}.b + \gamma.a)\,)\backslash\gamma \qquad \text{and} \qquad (l::a \mid k::b)$$

Here, depending on which summand is chosen in the left component of the first process, one needs to use the association $\varphi = \{(l,l),(k,k)\}$ or the association $\varphi' = \{(l,k),(k,l)\}$ (note that $\varphi \cup \varphi'$ is not consistent). Another example is given in the introduction.

To dynamically build up associations, we use the same technique as in [Ace91]. Let Φ be the set of consistent location associations. We define particular Φ-indexed families of relations S_φ over distributed processes, which we call *progressive bisimulation families* (although the relations that constitute a family are not themselves bisimulations). The idea is to start with the empty association of locations and extend it consistently as the bisimulation proceeds.

Definition 3.4 A *progressive bisimulation family (pbf)* is a Φ-indexed family $\mathbf{S} = \{S_\varphi \mid \varphi \in \Phi\}$ of relations over DIS, such that if $pS_\varphi q$ then for all $a \in Act, u \in Loc^*$:

(1) $p \overset{a}{\underset{u}{\Longrightarrow_s}} p' \;\Rightarrow\; \exists q', v$ such that $q \overset{a}{\underset{v}{\Longrightarrow_s}} q'$, $\varphi \cup \{(u,v)\} \in \Phi$ and $p' \,S_{\varphi\cup\{(u,v)\}}\, q'$

(2) $q \overset{a}{\underset{v}{\Longrightarrow_s}} q' \;\Rightarrow\; \exists p', u$ such that $p \overset{a}{\underset{u}{\Longrightarrow_s}} p'$, $\varphi \cup \{(u,v)\} \in \Phi$ and $p' \,S_{\varphi\cup\{(u,v)\}}\, q'$

(3) $p \overset{\tau}{\Longrightarrow_s} p' \;\Rightarrow\; \exists q'$ such that $q \overset{\tau}{\Longrightarrow_s} q'$ and $p' S_\varphi q'$

(4) $q \overset{\tau}{\Longrightarrow_s} q' \;\Rightarrow\; \exists p'$ such that $p \overset{\tau}{\Longrightarrow_s} p'$ and $p' S_\varphi q'$

Using these progressive bisimulation families, we may now define the *location equivalence* $\approx_\ell^s$ on CCS terms as follows:

Definition 3.5 (Static location equivalence) For $p,q \in$ CCS, we let $p \approx_\ell^s q$ if and only if for some $\bar{p},\bar{q} \in$ DIS such that $p\,\mathcal{D}\,\bar{p}$ and $q\,\mathcal{D}\,\bar{q}$, there exists a progressive bisimulation family $\mathbf{S} = \{S_\varphi \mid \varphi \in \Phi\}$ such that $\bar{p}\, S_\emptyset\, \bar{q}$.

The reader may have noticed that the inverse $\mathcal{D}^{-1}$ of the distribution relation is a function. If we let $\pi =_{\text{def}} \mathcal{D}^{-1}$, then $\pi(p)$ gives the CCS process underlying the distributed process p. It may be easily shown that all distributions of the same process are in the relation $S_\emptyset$ for some progressive bisimulation family $\mathbf{S}$:

Proposition 3.6 *Let* $p_1, p_2 \in \mathrm{DIS}$. *Then* $\pi(p_1) = \pi(p_2) \;\Rightarrow\; \exists$ *pbf* S *s.t.* $p_1 S_\emptyset p_2$.

Using this fact, we may prove that $\approx^s_\ell$ is indeed an equivalence relation and that, moreover, it is independent from the particular distributions that are chosen.

Proposition 3.7 (Properties of $\approx^s_\ell$)

1. *The relation* $\approx^s_\ell$ *is an equivalence on CCS processes.*
2. *For any* $p, q \in \mathrm{CCS}$: $p \approx^s_\ell q \;\Leftrightarrow$ *for all* $\bar{p}, \bar{q} \in \mathrm{DIS}$ *such that* $p \mathcal{D} \bar{p}$ *and* $q \mathcal{D} \bar{q}$ *there exists a progressive bisimulation family* $\mathrm{S} = \{S_\varphi \mid \varphi \in \Phi\}$ *such that* $\bar{p} S_\emptyset \bar{q}$.

Thus to check the equivalence of CCS processes we may pick arbitrary distributions of them. By virtue of this result, we can restrict our attention to particular "binary" distributions, systematically associating location 0 to the left operand and location 1 to the right operand of a parallel composition. A distribution of this kind will be called *canonical*, and elements of $\{0,1\}^*$ will be called *canonical locations*. These are exactly the locations used in [MN92] and, with a slightly different notation, in [MY92],[Yan93].

Let us see now a simple example, which shows the difference between location equivalence and causality-based equivalences, such as the causal bisimulation of [DD90]:

Example 3.8 $a.b + b.a \;\not\approx^s_\ell\; (a.\gamma \mid \bar{\gamma}.b)\backslash\gamma \;+\; (b.\gamma \mid \bar{\gamma}.a)\backslash\gamma \;\approx^s_\ell\; a \mid b$

As we announced earlier, $\approx^s_\ell$ will be shown to coincide with the dynamic equivalence $\approx^d_\ell$ of [BCHK91b]. Therefore all the examples given there for $\approx^d_\ell$ apply to $\approx^s_\ell$ as well.

3.2 Static location preorder

We define now a preorder $\sqsubseteq^s_\ell$ on CCS processes, which formalises the idea that one process is more sequential or *less distributed* than another. This preorder is obtained by slightly relaxing the notion of consistent association. The intuition for $p \sqsubseteq^s_\ell q$ is that there exist two distributions $\bar{p}$ and $\bar{q}$ of them such that whenever $\bar{p}$ can perform two transitions at independent locations, then $\bar{q}$ performs corresponding transitions at locations which are also independent, while the reverse is not necessarily true. This is expressed by the following notion of left-consistency:

Definition 3.9 A relation $\varphi \subseteq (Loc^* \times Loc^*)$ is a *left-consistent location association* if:

$$(u, v) \in \varphi \;\;\&\;\; (u', v') \in \varphi \;\;\Rightarrow\;\; (u \diamond u' \;\Rightarrow\; v \diamond v')$$

Now, if Ψ is the set of left-consistent location associations, we may obtain a notion of *progressive pre-bisimulation family (ppbf)* on distributed processes of DIS by simply replacing Φ by Ψ in Definition 3.4. Again, this gives rise to a relation on CCS processes:

Definition 3.10 (Static location preorder) If $p, q \in \mathrm{CCS}$, let $p \sqsubseteq^s_\ell q$ if and only if for some $\bar{p}, \bar{q} \in \mathrm{DIS}$ such that $p \mathcal{D} \bar{p}$ and $q \mathcal{D} \bar{q}$, there exists a progressive pre-bisimulation family $\mathrm{S} = \{S_\psi \mid \psi \in \Psi\}$ such that $\bar{p} S_\emptyset \bar{q}$.

It is easy to see that $p \approx^s_\ell q \;\Rightarrow p \sqsubseteq^s_\ell q$. As may be expected the reverse is not true. We have for instance, for the processes of Example 3.8 above:

Example 3.11 $a.b + b.a \;\sqsubseteq^s_\ell\; (a.\gamma \mid \bar{\gamma}.b)\backslash\gamma \;+\; (b.\gamma \mid \bar{\gamma}.a)\backslash\gamma$

We shall show that this static preorder coincides with the dynamic location preorder $\sqsubseteq^d_\ell$ of [BCHK91b], and thus inherits the theory of the latter.

For each $\mu \in Act_\tau$, $u \in Loc^*$, let $\xrightarrow[u]{\mu}_s$ be the least relation $\xrightarrow[u]{\mu}$ on LCCS processes satisfying the following axiom and rules.

(S1)	$\mu.p \xrightarrow[\varepsilon]{\mu} p$		
(S2)	$p \xrightarrow[u]{\mu} p'$	$\Rightarrow$	$l :: p \xrightarrow[lu]{\mu} l :: p'$
(S3)	$p \xrightarrow[u]{\mu} p'$	$\Rightarrow$	$p \mid q \xrightarrow[u]{\mu} p' \mid q$ $q \mid p \xrightarrow[u]{\mu} q \mid p'$
(S4)	$p \xrightarrow[u]{\alpha} p',\ q \xrightarrow[v]{\bar{\alpha}} q'$	$\Rightarrow$	$p \mid q \xrightarrow[u \sqcap v]{\tau} p' \mid q'$
(S5)	$p \xrightarrow[u]{\mu} p'$	$\Rightarrow$	$p + q \xrightarrow[u]{\mu} p'$ $q + p \xrightarrow[u]{\mu} p'$
(S6)	$p \xrightarrow[u]{\mu} p',\ \mu \notin \{\alpha, \bar{\alpha}\}$	$\Rightarrow$	$p \backslash \alpha \xrightarrow[u]{\mu} p' \backslash \alpha$
(S7)	$p \xrightarrow[u]{\mu} p'$	$\Rightarrow$	$p\langle f \rangle \xrightarrow[u]{f(\mu)} p'\langle f \rangle$
(S8)	$p[rec\, x.\, p/x] \xrightarrow[u]{\mu} p'$	$\Rightarrow$	$rec\, x.\, p \xrightarrow[u]{\mu} p'$

Figure 1: Static location transitions

Let $p \xrightarrow[u]{\tau}_d q \Leftrightarrow_{\text{def}} p \xrightarrow[u]{\tau}_s q$, and for each $a \in Act$, $u \in L^*$, let $\xrightarrow[u]{a}_d$ be the least relation $\xrightarrow[u]{a}$ on LCCS processes satisfying rules (S2), (S3), (S5), (S6), (S7), (S8) and the axiom:

(D1) $\quad a.p \xrightarrow[l]{a} l :: p \qquad$ for any $l \in Loc$

Figure 2: Dynamic location transitions

---

4 Dynamic approach

We briefly recall here the dynamic approach of [BCHK91b], and in particular the definitions of $\approx_\ell^d$ and $\sqsubseteq_\ell^d$. In the dynamic approach, locations are associated with actions rather than with parallel components. This association is built dynamically, according to the rule:

(D1) $\quad a.p \xrightarrow[l]{a}_d l::p \quad$ for any $l \in Loc$

In some sense locations are transmitted from transitions to processes, whereas in the static case we had the inverse situation. Rule (D1) is the essence of the dynamic location semantics. The remaining rules are just as in the static semantics, see Figure 2. We refer to [BCHK91b] for more intuition on the dynamic notion of location: let us just observe that these locations increase at each step, even if the execution goes on within the same parallel component. In fact the location l which appears in rule (D1) may be seen as an identifier for the action a. Then the location u of a generic transition $p \xrightarrow[u]{a}_d p'$ is a record of all the actions which causally precede a, what we shall call also the *access path* to a.

Because of rule (D1), the dynamic location transition system is both infinitely branching and infinitely progressing: it gives infinite representations for all regular processes. Indeed, this has been the main criticism to this dynamic approach, see [Ace91],[MY92],[MN92]. In fact, while the infinite branching may be overcome easily (through a *canonical* choice of dynamic locations, see [Cas93]) the infinite progression is really intrinsic to this semantics.

Note that for τ-transitions, for which we do not want to introduce additional locations, we simply use the static transition rules. Although this last point differentiates our strong dynamic location transition system from that originally introduced in [BCHK91b], the resulting *weak (dynamic) location transition system* is the same. The definition of the weak transitions $\stackrel{a}{\underset{u}{\Longrightarrow}}_d$ and $\stackrel{\tau}{\Longrightarrow}_d$ is similar to that of the $\stackrel{a}{\underset{u}{\Longrightarrow}}_s$ and $\stackrel{\tau}{\Longrightarrow}_s$.

We define now the *dynamic location equivalence* $\approx_\ell^d$ and the *dynamic location preorder* $\sqsubseteq_\ell^d$. Because of the flexibility in the choice of locations, these definitions are much simpler than in the static case. In [BCHK91b] the relations $\approx_\ell^d$ and $\sqsubseteq_\ell^d$ are obtained as instances of a general notion of *parameterized location bisimulation*. We shall use here directly the instantiated definitions.

Definition 4.1 (**Dynamic location equivalence**) A symmetric relation $R \subseteq \text{LCCS} \times \text{LCCS}$ is called a *dynamic location bisimulation (dlb)* iff for all $(p,q) \in R$ and for all $a \in Act, u \in Loc^+$:

(1) $p \stackrel{a}{\underset{u}{\Longrightarrow}}_d p' \;\Rightarrow\; \exists q' \in \text{LCCS}$ such that $q \stackrel{a}{\underset{u}{\Longrightarrow}}_d q'$ and $(p',q') \in R$

(2) $p \stackrel{\tau}{\Longrightarrow}_d p' \;\Rightarrow\; \exists q' \in \text{LCCS}$ such that $q \stackrel{\tau}{\Longrightarrow}_d q'$ and $(p',q') \in R$

The largest *dlb* is called *dynamic location equivalence* and denoted $\approx_\ell^d$.

We refer to [BCHK91b] for examples and results concerning $\approx_\ell^d$. Consider now the location preorder $\sqsubseteq_\ell^d$. Here, instead of requiring the identity of locations in corresponding transitions, we demand that the locations in the second (more distributed) process be subwords of the locations in the first (more sequential) process. Formally, the *subword* relation $\leq_{\text{sub}}$ on Loc^* is defined by: $v \leq_{\text{sub}} u \Leftrightarrow \exists v_1,\ldots,v_k,\ \exists w_1,\ldots,w_{k+1}$ s.t. $v = v_1 \cdots v_k$ and $u = w_1 v_1 \cdots w_k v_k w_{k+1}$.

Definition 4.2 (Dynamic location preorder) A relation $R \subseteq$ LCCS $\times$ LCCS is called a *dynamic location pre-bisimulation (dlpb)* iff for all $(p,q) \in R$ and for all $a \in Act, u \in Loc^+$:

(1) $p \stackrel{a}{\underset{u}{\Longrightarrow}}_d p' \Rightarrow \exists v \leq_{\text{sub}} u, \exists q' \in$ LCCS such that $q \stackrel{a}{\underset{v}{\Longrightarrow}}_d q'$ and $(p',q') \in R$

(2) $q \stackrel{a}{\underset{v}{\Longrightarrow}}_d q' \Rightarrow \exists u.\, v \leq_{\text{sub}} u, \exists p' \in$ LCCS such that $p \stackrel{a}{\underset{u}{\Longrightarrow}}_d p'$ and $(p',q') \in R$

(3) $p \stackrel{\tau}{\Longrightarrow}_d p \Rightarrow \exists q' \in$ LCCS such that $q \stackrel{\tau}{\Longrightarrow}_d q'$ and $(p',q') \in R$

(4) $q \stackrel{\tau}{\Longrightarrow}_d q' \Rightarrow \exists p' \in$ LCCS such that $p \stackrel{\tau}{\Longrightarrow}_d p'$ and $(p',q') \in R$

The largest *dlpb* is called *dynamic location preorder* and denoted $\sqsubseteq^d_{\approx\ell}$.

The intuition is as follows. If p is a sequentialized version of q, then each component of p corresponds to a group of parallel components in q. Then the local causes of any action of q will correspond to a subset of local causes of the corresponding action of p. This may be easily verified for the following examples:

Example 4.3 $\quad a.a.a \;\sqsubseteq^d_{\approx\ell}\; a.a \mid a \qquad$ and $\qquad a.b + b.a \;\sqsubseteq^d_{\approx\ell}\; a \mid b$

We shall not comment further here on the relations $\approx^d_\ell$ and $\sqsubseteq^d_{\approx\ell}$, referring again the reader to [BCHK91b] for more examples and for results concerning these relations. We proceed now to state our main result, namely that the dynamic relations $\approx^d_\ell$ and $\sqsubseteq^d_{\approx\ell}$ coincide with the static relations $\approx^s_\ell$ and $\sqsubseteq^s_{\approx\ell}$ introduced in the previous section.

Theorem 4.4 *Let $p,q \in$ CCS. Then: $p \approx^s_\ell q \Leftrightarrow p \approx^d_\ell q$ and $p \sqsubseteq^s_{\approx\ell} q \Leftrightarrow p \sqsubseteq^d_{\approx\ell} q$.*

To prove this results, we use a new transition system on CCS, called *occurrence system*, which is essentially a simplification of the *event system* introduced in [BC91] to compare different models of CCS. This transition system, which incorporates the information of both location transition systems, is used as an intermediate between the static and the dynamic semantics. The main point is to prove that starting from a static or a dynamic location computation, one may always reconstruct a corresponding occurrence computation. This means, essentially, that all the information about distribution and local causality is already present in both location transition systems. The proof may be found in [Cas93].

Acknowledgements

The idea of a static assignment of locations was originally put forward by G. Boudol in the course of a CEDISYS meeting in Brighton, in September 1990. I would like to thank him for inspiration and for innumerable comments and advices. I also benefitted from discussions with L. Aceto, who was the first to formalise the "static view of locations" for a subset of CCS. I am grateful to P.S. Thiagarajan, for commenting on an earlier draft of this paper and for raising several interesting questions. Part of this work was done while visiting Thiagarajan and his colleagues in Madras. I would like to thank all of them for their interest and comments.

References

[Ace91] L. Aceto. A static view of localities. Report 1483, INRIA, 1991. To appear in *Formal Aspects of Computing.*

[BC91] G. Boudol and I. Castellani. Flow models of distributed computations: three equivalent semantics for CCS. Report 1484, INRIA, 1991. To appear in *Information and Computation.* Previous version in Proc. La Roche-Posay, LNCS 469, 1990.

[BCHK91a] G. Boudol, I. Castellani, M. Hennessy, and A. Kiehn. Observing localities. Report 4/91, Sussex University, and INRIA Res. Rep. 1485, 1991. To appear in *Theoretical Computer Science.* Extended abstract in Proc. MFCS 91, LNCS 520, 1991.

[BCHK91b] G. Boudol, I. Castellani, M. Hennessy, and A. Kiehn. A theory of processes with localities. Report 1632, INRIA, 1991. To appear in *Formal Aspects of Computing.* Extended abstract in Proc. CONCUR92, LNCS 630, 1992.

[Cas93] I. Castellani. Full version of this paper. Report, INRIA, 1993. To appear.

[DD90] Ph. Darondeau and P. Degano. Causal trees: interleaving + causality. In *Proceedings LITP Spring School, La Roche-Posay*, number 469 in LNCS, 1990.

[DDNM87] P. Degano, R. De Nicola, and U. Montanari. Observational equivalences for concurrency models. In M. Wirsing, editor, *Formal Description of Programming Concepts–III, Proceedings of the* 3^{th} *IFIP WG 2.2 working conference,* Ebberup 1986, pages 105–129. North-Holland, 1987.

[GG89] R.J. van Glabbeek and U. Goltz. Equivalence notions for concurrent systems and refinement of actions. Arbeitspapiere der GMD 366, Gesellschaft für Mathematik und Datenverarbeitung, Sankt Augustin, 1989. Extended abstract in Proc. MFCS 89, LNCS 379, 1989.

[Kie91] A. Kiehn. Local and global causes. Report 342/23/91, Technische Universität München, 1991. Submitted for publication.

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

[MN92] M. Mukund and M. Nielsen. CCS, locations and asynchronous transition systems. In *Proceedings FST-TCS 92*, 1992.

[MY92] U. Montanari and D. Yankelevich. A parametric approach to localities. In *Proceedings ICALP 92*, number 623 in LNCS, 1992.

[Yan93] D. Yankelevich. *Parametric Views of Process Description Languages.* Ph.d. thesis, University of Pisa, 1993.