4

Using PO Methods for Verifying Behavioural Equivalences

Monica Lara de Souza
Robert de Simone
INRIA
INRIA, BP 93, 06902 Sophia-Antipolis, FRANCE. Telephone: (33) 93 95 74 92, Fax: (33) 93 95 74 88. e-mail:
mlsouza@sophia.inria.fr,rs@sophia.inria.fr

Abstract

The modelling of concurrent systems by synchronized distributed automata generates naturally an independence notion between the events of the global system. The theory of (Mazurkiewicz) traces suggests that such an independence relation induces a nice equivalence relation over the sequences of events of the system. Two sequences will be equivalent just in case they constitute two different interleavings of the same stretch of partially ordered behavior.

We want to use reductions of the interleaved global system by this independence relation for verification means. The allowed reduction depends on the property to conserve, the prime condition being to retain at least one representative for each trace. Others recent published works on these methods (often called *Partial-Order methods*) have proposed the checking of particular linear temporal properties on these reduced models. We study it here in the context of equivalence checking, where a concurrent process description is confronted to a specification by comparing their labeled transition systems together (relative to *bisimulation* for instance). We define a reduced global automaton with predicates on the states where transitions were suppressed, and conditions for retaining enough information for representing the global system for verification. The limits of the approach are shown as well as suggestions for practical applications and some experimental results on classical examples.

Keywords

Verification, Bisimulation, Partial-Order Methods, Automata Network, Reduction

1 INTRODUCTION

Modelling of (finite state) concurrent systems has led in the past to a rich body of automatic verification techniques. The main problem is generally the potential state explosion of the underlying global system. Many methods have been proposed to cope with this, each with its own intrinsic qualities and limitations. In this paper we shall focus on so-called *Partial-Order methods* (Wolper and Godefroid 1993), (Peled 1994), (Valmari 1990), where one tries to discard

G. V. Bochmann et al. (eds.), *Formal Description Techniques VIII*

redundant interleavings of independent behaviours off the global automaton, provided ability to check some given property is retained. We shall call *partial automaton* such an incomplete automaton. The problem lays with confluent *diamonds*, or how two independent actions can permute and be performed in any order. The goal is to remove edges from these diamonds -and then further disconnected vertices-, in a semantically non-destructive fashion; technically, each remaining trace should in particular have an extension being represented by *at least* one of its linearisations.

Most research in partial-order methods has been so far devoted to the verification of linear temporal specifications, which models the execution of a program as computation sequences. An extensive bibliographic survey is provided in (Wolper and Godefroid 1993). Here we will rather focus on general notions of operational equivalences, like *bisimulation* or yet *trace language equality*. We focus on abstract requirements on partial automata to preserve bisimulation checking abilities. These requirements characterise the limits of reduction (in transitions and target states), so that enough information is left for a natural definition of bisimulation relations.

A transition can be discarded for any of two reasons: either as it was already *anticipated* (in the course of an alternative path), or as it can be *delayed* further (down some ongoing path). This corresponds to dynamic notions of *sleep sets* and *persistent sets* in (Wolper and Godefroid 1993). Our axiomatic conditions will rely heavily on explicit *Pre/Post* predicates on *partial states*, which recall 'shadows" of omitted transitions together with the reason why they were disposed of. This reintroduces only part of the larger automaton, since target states beyond are still potentially left out as unreachable and therefore unrepresented.

The paper is structured as follows: in section 2, after some basic definitions, we provide the definition of a *complete automaton* for a given independence relation. The global automaton underlying the *synchronized vectors of automata* is *complete* for the independence relation naturally deduced from the distribution.

In section 3 we introduce *partial* –or incomplete– automata. The definition includes axiomatic requirements on partial states enforcing "faithfulness", in that reduction should not be excessive. It refers explicitly to the underlying complete automaton. We show this reference to be necessary through an example where the same partial automaton can be obtained by reduction of non-bisimilar complete forms. We describe also a generic counterexample for the completeness of partialization with respect to bisimulation of the relied complete counterparts. We show two bisimilar complete automata, one of which allows a partial form which cannot be matched by the other through reduction. The problem resides in the fact that partial reductions (with anticipated behaviours) deal with actual states and their identities, while plain bisimulation is only concerned with equivalent states.

In section 4 we consider behavioural relations amongst partial automata: language containment and equality, (adapted) simulation refinement and bisimulation equivalence. We show that language containment (equality) of partial automata imply the corresponding relation on complete ones. Concerning partial bisimulation of partial automata, we show that it implies language equivalence of complete forms, and also bisimulation in the case where only deterministic behaviours can be omitted (while preserved parts can be nondetermistic still). At the end of the section we propose some reductions on τ-transitions.

In section 5 we discuss the implementation of the ideas exposed, and show the obtained results on the classical Milner's scheduler.

2 COMPLETE AUTOMATA

This section recalls very basic definitions about partially commutative (Mazurkiewicz) traces and related automata structures. We extend some of their characterisations with confluence properties on non-deterministic automata, thereby introducing complete automata as a new model (complete here reads as: *relative to a given independence relation*).

2.1 Basic definitions

Automata

Notations 1 (Automata) *An* automaton A *is a 4-uple* $< L_A, Q_A, init_A, T_A >$, *with:*

- L_A *a finite set of* action labels,
- Q_A *a finite set of* states,
- $init_A \in Q_A$ *an* initial state *and*
- $T_A \subset (Q_A \times L_A \times Q_A)$ *a behavioural* labeled transition relation.

We also note $Sort(A)$ *for* L_A, $p \xrightarrow{a}_A q$ *for* $(p, a, q) \in T_A$, *and extend the transitions to sequences of actions.*

We let $Now_A = \{(p, a) / \exists q, (p, a, q) \in T_A\}$.
We drop A *subscripts when clear from context.*

Synchronised vectors of automata (or: networks)

A specification model often used in verification theory is the *Synchronised Vector of Automata*. It consists of a vector of automata, each endowed with a *sort* describing the actions it is involved in, so that synchronisation and communication simply consist in that: *an action is performed globally iff it is performed simultaneously by all components which have it in their sorts.*

Formally:

Definition 1 Synchronised Vector of Automata *A* Synchronised Vector of Automata V *is simply an ordered list* $(A_1, \ldots, A_n)$ *of automata.*

Notations 2 *Given* $a \in \bigcup_{1 \leq i \leq n} L_{A_i}$ *we note* $Loc(a)$ *(for "locations of* a*") the set* $\{i, a \in L_{A_i}\}$.

In the sequel an *event* shall be a set of local transitions which may perform an action together, disregarding local states at other unconcerned locations. Thus an event is somehow a more concrete form of action, recalling which actual local transitions were effectively used in it.

Global System derived from the network

There is an obvious definition for the global automaton obtained from a given synchronised vector of automata, based on cartesian product:

Definition 2 Global expansion of a SVA *Given* V *as above, we note* V *the automaton defined by:*

- $L_{\bar{V}} = \bigcup_{1 \leq i \leq n} L_{A_i}$;
- $Q_{\bar{V}} = Q_{A_1} \times \ldots \times Q_{A_n}$;
- $q^0_{\bar{V}} = (q^0_{A_1}, \ldots, q^0_{A_n})$;
- $T^a_{\bar{V}} = \{(q_{A_1}, \ldots, q_{A_n}), a, (q'_{A_1}, \ldots, q'_{A_n}))/\forall i \in Loc(a), (q_{A_i}, a, q'_{A_i}) \in T_{A_i}, \forall i \notin Loc(a), q_{A_i} = q'_{A_i}\}$.

Such global automata are usually restricted to reachable parts.

Independence

This model generates naturally an independence notion between the actions of the global system.

Two actions a and b are said to be *independent* if $Loc(a) \cap Loc(b) = \emptyset$. We note $a \smile b$ then. We let $\smile_V$ be the whole symmetric relation of independent couples. We extend the notation to sequences so that $s \smile_V s'$ when each individual letter from s is independent from all letters in s'.

Traces

The theory of Mazurkiewicz (Mazurkiewicz 1986) traces suggests that such an independence relation induces a nice equivalence relation over the sequences.

Definition 3 Traces *Given $s \in L^\star$ a sequence of actions and $\smile$ a symmetric irreflexive relation (to be thought of as a permutability relation), we note $\hat{s}$ the equivalence class of s generated by successive permutations of related actions, and call it the* trace *associated with s. We note $\hat{L}$ the set of traces on L.*

2.2 Complete Automaton

Definition 4 Complete Automaton

Let $\smile \subseteq L_A \times L_A$ be a binary symmetric irreflexive relation (representing permutability by independence). $(A, \smile)$ is a -possibly non-deterministic- complete automaton *if* $\forall (a,b) \in \smile$*:*

- $\forall p, p', q' \in Q, \left(p \xrightarrow{a} p' \xrightarrow{b} q'\right) \Rightarrow \left(\exists\, !\, q \in Q, p \xrightarrow{b} q \xrightarrow{a} q'\right)$
- $\forall p, p', q \in Q, \left(p \xrightarrow{a} p' \wedge p \xrightarrow{b} q\right) \Rightarrow \left(\exists\, !\, q' \in Q, p' \xrightarrow{b} q' \wedge q \xrightarrow{a} q'\right)$
- $\forall (a,c) \in \smile, \left(p \xrightarrow{a} p'\right) \Rightarrow \left(\left(\exists q,\ p \xrightarrow{b} q \wedge p \xrightarrow{c} q\right) \Leftrightarrow \left(\left(\exists q',\ p' \xrightarrow{b} q' \wedge p' \xrightarrow{c} q'\right)\right)\right)$

Furthermore we shall assume that all states are reachable (from the initial one). Note the uniqueness requirement on q (resp. q') in the first (resp. second) condition.

The completeness assumptions are depicted graphically in figure 1 and 2. In the deterministic case complete automata are also called *asynchronous transition systems* in (Winskel and Nielsen 1993), following (Bednarczyk 1988) and (Shields 1985).

As recalled in the trivial next lemma, only the first axiom –and even without the unicity requirement– is strictly needed to ensure existence of all representatives of a given (partially

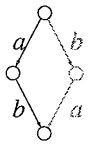

Figure 1 Permutation.

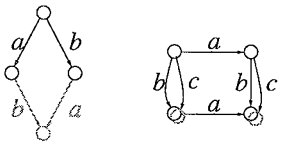

Figure 2 Confluence/independence.

commutative) trace, so that $\smile$ is really a permutation relation. Other conditions state that $\smile$ is an independence relation, by adding confluence requirements on actions with disjoint effects.

Conversely, given *any* automaton one can easily compute a largest permutation relation it is complete for, since the union of independence relations shall remain an independence relation.

Lemma 1 *Let A be complete with respect to $\smile$. Suppose $p \xrightarrow{s} q, s \in L^\star$. Then $\forall s'$ such that $\widehat{s} = \widehat{s'}$, $p \xrightarrow{s'} q$.*

We note $\mathcal{L}_A(p)$ the (partially commutative) language recognized by A rooted in p.

Fact 1 *Let $FanOut(p,a)$ be the number of states reached from p through a. Let $b \smile a$, and $p \xrightarrow{b} p'$. Then $FanOut(p,a) = FanOut(p',a)$*

Fact 2 $\bar{V}$ *is* complete *relative to* $\smile_V$.

We can also adapt the definition of *Complete Automata* to events as labels, instead of mere action names. Then obviously the transition system becomes deterministic.

3 PARTIAL AUTOMATA

We now consider the problem of omitting some possibly redundant transitions from a complete automaton. This should produce a *partial*, incomplete automaton, with far less reachable states in general.

Our general goal is to verify behavioural equivalences of complete forms on the -smaller- partial ones. Obviously one must then be guarded against excessive reductions, and ensure that partial automata remain basically faithful to their complete counterparts. We shall attempt to propose such criteria on partial automata.

These axiomatic conditions will be phrased using auxiliary predicates Pre and $Post$, which provide a record of "shadow" transitions respectively discarded as *anticipated* or *postponed*. Some such conditions are *internal* in that they deal only with the structure of the partial automaton, and one could compute the largest $Pre/Post$ predicates satisfying them. Other conditions are linking behaviours from the "partial states" to those in the underlying complete automaton. So, a partial automaton is **not** defined in isolation, irrespective of its underlying complete gen-

erator. We shall in fact run into examples where, due to non-determinism, isomorphic partial automata were obtained from non-bisimilar complete forms.

On the other hand, reduction functions will attempt **not to** build the global complete automaton, but rather keep it virtual and find approximation methods that prove sufficient to enforce our requirements. In this respect new approximation techniques can be tried against these requirements.

3.1 Common facts on reduction functions

We briefly recall some essential concerns of existing reduction procedures, only to motivate our introduction of $Pre/Post$ sets. Let us consider the basic *diamond* situation of figure 3 (here

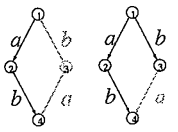

Figure 3 Simplifications.

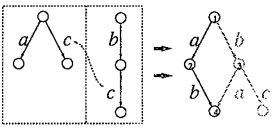

Figure 4 Wrong choice.

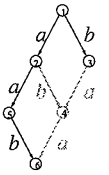

Figure 5 Combined gain.

$a \smile b$). Trying to discard transitions, one should of course retain *at least* one for each label. Then two "obvious" kinds of saving can be considered: keeping states $\{1,2,3\}$ or $\{1,2,4\}$ (or symmetrically $\{1,3,4\}$). But:

- keeping $\{1,2,3\}$ makes unreachable state 4, without means of recovering its following behaviours. Therefore one must then retain either of the two lower transitions ($2 \rightarrow 4$ or $3 \rightarrow 4$). This reduction saves only one transition, and no state. It should be noted that the transition is safely omitted because the corresponding behaviour was already performed earlier at a **Pre**vious stage.
- keeping $\{1,2,4\}$ (or $\{1,3,4\}$) will provoke a similar problem if the removed state had transitions which now become unreachable. But this possibility can be exactly characterised, so that the reduction rule is used only when sound. Figure 4 gives a generic (minimal) example of unsound omission, where a third action, enabled as a consequence of b only, can "steal" a. Effectively deciding soundness of the reduction of course involves reaching parts of the automaton beyond the omitted, exactly what reduction functions try to avoid. Therefore the crux of all these techniques lays in the definition of sufficient static approximations avoiding this search.

It should be noted that in the second type of reduction the early transition can be omitted because it can be **Post**poned later.

Figure 5 shows how the two type of reductions can be combined to save as many states as possible, when the first type allows not to reintroduce states already saved by the second type of reduction.

3.2 Faithfulness requirements

As seen in the previous section, a transition can be discarded for any of two reasons: either as it was already *anticipated* (in the course of an alternative path), or as it can be *delayed* further (down some ongoing path). This corresponds to dynamic notions of *sleep sets* and *persistent sets* in (Wolper and Godefroid 1993).

We now introduce *partial automata* as deduced from complete ones through transition omissions, recalled by corresponding predicates ($Pre/Post$). Partial automata need to fulfill requirements, based on these predicates, to prove relatively faithful to their complete form originators.

Notations 3 *Let A and B be two automata. We note $A \subset B$ when: $Q_A \subset Q_B, L_A = L_B$, $init_A = init_B$, $T_A \subset T_B$.*

We shall use the case when B is complete, while not necessarily A.

Definition 5 *Let B be a complete automaton w.r.t. an independence relation $\smile$. We call* partial automaton *(associated to $(B, \smile)$) a structure $(A, \smile, Pre_A, Post_A)$, where $A \subset B$ and $Pre, Post \subseteq (Q_A \times L_A)$ are two predicates (or equivalently two functions $Q_A \to 2^{L_A}$), satisfying the following requirements:*

1. Now_A, Pre_A, $Post_A$ *are pairwise disjoint, and* $\forall a,\ (init, a) \notin Pre_A$,
2. $\forall a, b, p, q, r,$
   $\Big(a \smile b \wedge (p \xrightarrow{a}_A q) \wedge (p \xrightarrow{b}_A r)\Big) \Rightarrow (b \in Pre_A(q) \Rightarrow (Pre_A(r) \cap \smile_a) \subset (Pre_A(q) \cap \smile_b))$
3. $(p, a) \in Pre_A \Longrightarrow \Big(\forall b, q, (q \xrightarrow{b}_A p) \Rightarrow (a \smile b \wedge (q, a) \in Pre_A \cup Now_A)\Big)$
4. $(p, a) \in Pre_A \Longrightarrow \Big(\exists s, q,\ (q \xrightarrow{s}_A p) \wedge a \smile s \wedge (q, a) \in Now_A\Big)$
5. $(p, a) \in Post \Longrightarrow \Big(\forall b, q, (p \xrightarrow{b}_A q) \Rightarrow (a \not\smile b \vee (q, a) \in Post_A \cup Now_A)\Big)$
6. $(p, a) \in Post_A \Longrightarrow \Big(\exists s, q,\ (p \xrightarrow{s}_A q) \wedge a \smile s \wedge (q, a) \in Now_A\Big)$
7. $p \xrightarrow{a}_B q \Leftrightarrow (p, a) \in (Pre_A \cup Post_A) \vee p \xrightarrow{a}_A q,$
8. $(p, a) \in Post_A, \forall b \in Now_A(p), \forall s \in L_A^\star,$
   $\Big(\exists c \in L,\ (a.s) \smile b \ \wedge\ c \not\smile b \ \wedge\ p \xrightarrow{a.s.c}_B\Big) \Longrightarrow \Big((a.s) \smile c \wedge p \xrightarrow{c}\Big)$

We note Pre_A^p for $\{a, (p, a) \in Pre_A\}$ (and likewise for Now and $Post$).

We now comment on some of these axiomatic requirements.
All first six axioms are technical conditions on well-behaving of predicates:

- they should be disjoint (axiom 1);
- two independent actions should not mutually discharge the requirement of continuation to one another (axiom 2);
- an anticipated behaviour should indeed be taken into account in all previous states (axiom 3); similarly a postponed behaviour should be taken into account in all successor states reached through actions that are independent with the concerned one (axiom 5);
- an anticipated behaviour should always be retrieved in a finite backward run (axiom 4);
- similarly, postponed behaviours should always be retrieved in a finite forward run (axiom 6).

The seventh condition simply states that behaviours in B have to be dealt with in A (and not possibly forgotten). The more complex eighth condition gives full generality to the counterexample in figure 4. It says that, whenever action a is delayed through b, it could not cause disabling of b in B other than those already present in the current state in A.

So it appears that the last two axioms are in essence different from the six previous ones, as they link behaviours of automata A and B together. In practice one could construct two "largest" predicates Pre and $Post$ satisfying the first set of axioms only from the transitions in A, and then what is really to be verified is that such predicates would validate the two last axioms, dealing with relations to underlying B.

3.3 Adequacy problems for partial automata

We claim the axioms above to be highly natural for a definition of partial automata. Still it remains to be seen whether partial automata could be characterised in isolation: is it the case that all complete automata allowing *the same* partial form are strongly related? The counterexample of figure 6 provides a negative answer: the two nets produce nonbisimilar complete automata, but which allow the same partial one. It should still be noted that complete forms have here the same trace language.

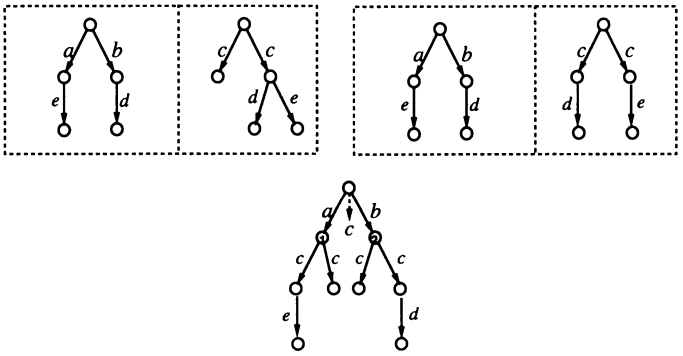

Figure 6 Nonbisimilar complete with identical partial automata.

In the next section we shall look for relations on couples of partial automata ensuring that the underlying complete automata have identical trace languages, or even are strongly bisimilar (the latter will require "some" determinacy assumptions).

3.4 Completeness problems for partial automata

The previous section dealt with questions of adequacy (for partial bisimulation on partial automata). Regarding *completeness*, one can ask whether, given two bisimilar complete automata and a partial reduction of one, the other can be reduced so as to be equivalent to it. The answer

shall be negative. The notion of partial reduction with anticipated (Pre) behaviours is mostly sensible to state identity, while bisimulation only cares to states *up to* equivalence.

Figures 7, 8 exemplify this (here upward dashed arrows mark Pre sets, downward dashed arrows mark $Post$ sets). The only difference in the distributed descriptions is that two (equivalent) states in the middle component are being merged into a single one. As a result the saving which was possible in global state 2 on the left, and not in state 1, is impossible altogether in state 0 on the right (to get such a gain, one should be able to anticipate a through all incoming behaviours at 0). It seems hopeless to figure a bisimulation-like notion which would relate state 2 to a state on the right and help prove equivalence of the initial states through action c.

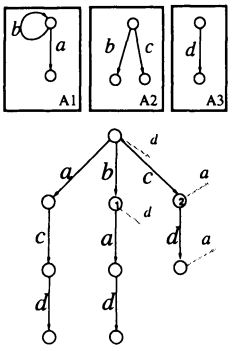

Figure 7 Bisimilar states.

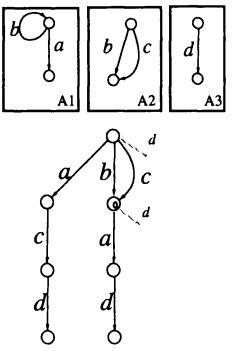

Figure 8 Single state.

This problem doesn't appear if omitted transitions correspond only to **Post**poned behaviours.

4 BEHAVIOURAL RELATIONS AMONGST PARTIAL AUTOMATA

4.1 (Finitary) Language inclusion

What is the exact property of partial automata relative to trace preservation? We state this in the following lemma, after defining auxiliary notations.

Notations 4 *Given* $(A, \smile)$ *a partial automaton and* D *a subset of* L_A*, we introduce the* initial discard *operator, noted* $/$*, on automata (states):*

$$\frac{p \xrightarrow{a} q \quad , \quad a \notin D}{p/D \xrightarrow{a} q/(D \cap I_a)}$$

where $I_a = \{b \in L_A, a \smile b\}$.

Initial discards shall be used to neglect the behavioural parts of complete automata which are declared as Pre anticipated behaviours, because nothing conclusive can be said on partial states

at this point (the information will be found in anterior states, on a backward path to the initial state).

Lemma 2 *Let A be a partial automaton, relative to complete $(B, \smile)$. Let $p \in Q_A$ and $s \in L_A^*$. Suppose $\exists q \in Q_B$, $p/(Pre_A^p) \xrightarrow{s}_B q$. Then $\exists t \in L_A^*$, $t \smile s$, $\exists p' \in Q_A$ such that $p \xrightarrow{\widehat{s.t}}_A p'$ and $q \xrightarrow{t}_B p'$.*

Proof. By induction on the length of s.
The case $s = \epsilon$ is trivial (take $t = \epsilon$, $p' = q$).
When $|s| > 1$, let $S = \{s_i = a_i.s_i' / \widehat{s_i} = \widehat{s}\}$. Obviously $\forall i, j,\ i \neq j,\ a_i \smile a_j$, and $\{a_i\} \cap Pre_A^p = \emptyset$ (otherwise we would not have $p/(Pre_A^p) \xrightarrow{s}_B$). Two cases may arise:

$S \not\subset Post_A^p$: Then one can prove the existence of i such that $\exists p_i,\ p \xrightarrow{a_i}_A p_i \xrightarrow{s_i'}_B q$, with $Pre_A^{p_i} \cap \{a_j\} = \emptyset$. The last condition is established through axiom 3, and causes p_i, s_i' to fulfill the proposition by induction. The t transition adopted for s is then the same as the one for s_i'.

$S \subset Post_A^p$: Then by a combination of axioms 7 and 8, there exists b, $b \smile a_i$ for all i, and p_b, q_b such that $p \xrightarrow{b}_A p_b \xrightarrow{s}_B q_b$ with $\{a_i\} \cap Pre_A^{p_b} = \emptyset$ as in the previous case (by axiom 4). This is the case where an extra behaviour is added to the potential t sequence of interspersed independent transitions. This step does not decrease the length of s, but as the number of states is finite, one can prove, using axiom 8, that it is taken a bounded number of times before a state meeting the first case is reached.

□

Our first main theorem now comes as a corollary of the previous lemma.

Theorem 1 *Let B and $B\prime$ be two complete automata (with identical independence relation), and A and $A\prime$ be two partial automata respectively associated to B and $B\prime$. Then $(\mathcal{L}(A) \subset \mathcal{L}(A\prime)) \Longrightarrow (\mathcal{L}(B) \subset \mathcal{L}(B\prime))$*

Proof. Let $s \in (\mathcal{L}(B) \setminus \mathcal{L}(B\prime))$. Then $\exists t,\ t \smile s,\ \widehat{s.t} \in \mathcal{L}(A)$, and so $\widehat{s.t} \in \mathcal{L}(A\prime) \subset \mathcal{L}(B\prime)$. But then, by permutation properties of $\mathcal{L}(B\prime)$, $s \in \mathcal{L}(B\prime)$. □

This theorem actually holds also for language equivalence, which is inclusion "both ways". It settles the problem for bisimulation also in the deterministic case, where both notions coincide.

4.2 Simulation refinement and bisimulation equivalence

We provide now definitions for simulation *refinement preorder* and bisimulation *equivalence* for partial automata.

Definition 6 (Partial Simulation Refinement)
Let A be a partial automaton. A binary relation $\prec \subseteq (A \times A)$ is a **P**-simulation refinement *if* $\forall p, q \in Q_A$, $p \prec q$ *implies:*

$$\begin{array}{lll} \forall a, q, & \left(p \xrightarrow{a}_A q\right) & \Rightarrow \left(\exists q' s.t.\ p' \xrightarrow{a}_A q' \wedge q \prec q'\right) \\ \forall a, & (Post(p,a)) & \Rightarrow (Post(p',a) \vee Now(p',a)) \\ \forall a, & (Pre(p,a)) & \Rightarrow (Pre(p',a) \vee Now(p',a)) \end{array}$$

This definition can be applied to a couple of automata with identical independence relations by forming the disjoint union of states.

Given A, the union of two refinements is again a refinement so that there is a largest refinement inside A. In general it is a preorder, as for instance all similar complete automata refine one another. In fact, refinement breaks down on complete automata to plain strong simulation.

Proposition 1

$$p \prec_A q \text{ (as states of } A) \Rightarrow \mathcal{L}(p/Pre_A^p) \subseteq \mathcal{L}(q/Pre_A^p) \text{ (as states of complete } B)$$

Proof. Trivial, since $(p \prec q) \Rightarrow (\mathcal{L}(p) \subset \mathcal{L}(q))$. □

4.3 (Partial) bisimulation equivalence

Definition 7 (Partial Bisimulation)
Let $(A, \smile, Pre_A, Post_A)$ be a partial automaton. A binary relation $\cong \subseteq (A \times A)$ is a **P**-bisimulation *if* $\forall p, q \in Q_A$, $p \cong q$ *implies:*

- $Pre_A^p = Pre_A^q$
- $Post_A^p = Post_A^q$
- $\forall a, p',\ p \xrightarrow{a} p' \Rightarrow \left(\exists q',\ q \xrightarrow{a} q' \wedge p' \cong q'\right)$
- $\forall a, q',\ q \xrightarrow{a} q' \Rightarrow \left(\exists p',\ p \xrightarrow{a} p' \wedge p' \cong q'\right)$

Again there is a largest such relation inside A, which is an equivalence.

Of course in presence of Pre sets states will *not* be compared according to these anticipated behaviours, locally undescribed. Actually we could have dropped condition $Pre_A^p = Pre_A^q$ (as well as $Pre_A^p \subset Pre_A^q$ in case of refinement) and got looser relations. Our choice was prompted by the will to obtain canonical minimisation notions which would preserve the axiomatics of partial automata.

Proposition 2

$$p \cong_A q \text{ (as states of } A) \Rightarrow \mathcal{L}(p/Pre_A^p) = \mathcal{L}(q/Pre_A^p)$$

Trivial, since $(p \cong q) \Rightarrow (p \prec q \wedge q \prec p)$.

This result is most intuitive when applied to (couples of) initial states, for which Pre sets are void.

Next, we introduce sufficient conditions so that **P**-bisimulation of partial forms induce bisimulation equivalence of corresponding complete forms. They will basically assert that all omitted

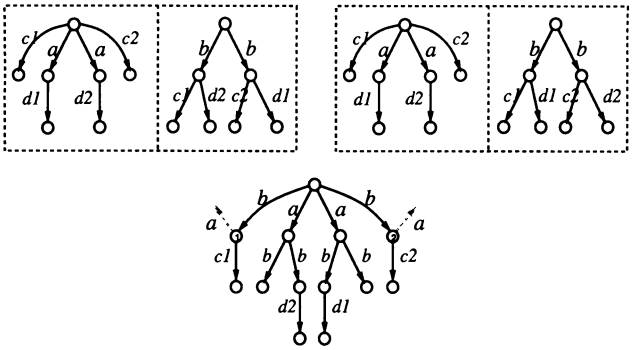

Figure 9 Partial bisimilar nets (with Pre actions) with non-bisimilar complete forms.

transitions should be *deterministic* in the complete form. The counterexample of figure 9 shows why we need this restriction on Pre sets also. * Again, the partial automata may look identical (as is displayed at the bottom of the picture), but the complete systems would not be equivalent, as dangling parts at states 1 and 2 are actually permuted.

Definition 8 *We let* $DetPre$ *be the predicate on* $Q_A \times L_A$ *defined as:*
$Pre(p,a) \Rightarrow \left(\forall b, \forall p',\ (p' \xrightarrow{b} p) \Rightarrow (FanOut_A(p',b) = 1)\right)$

The purpose of this definition is to ensure that an omitted Pre action is retrieved back through deterministic (back)actions, so that resetting actual transitions will be nonambiguous. Another counterexample, where a deterministic action (b) anticipated through nondeterministic (a) behaviours lead to bisimulation problems, is to be found in figure 10.

Definition 9 *We let* $DetPost$ *be the predicate on* $Q_A \times L_A$ *defined as:*
$Post(p,a) \Rightarrow \forall s \in L_A^\star,\ \forall b,c \in L_A,$
$$\left(\begin{array}{ll} \exists q,r, & q \neq r \wedge p \xrightarrow{a.s}_B q \wedge p \xrightarrow{a.s}_B r \\ \exists q',r', & q' \neq r' \wedge p \xrightarrow{b}_A q' \wedge p \xrightarrow{c}_A r' \end{array}\right) \Rightarrow (b \not\sim a.s \vee c \not\sim a.s)$$

Proposition 3 *Let* $\cong_A$ *be a* **P***-bisimulation amongst states of A and suppose in addition that* $\forall p \in Q_A, \forall a \in L_A, DetPre(p,a) \wedge DetPost(p,a)$.
Then, $\forall p,q \in Q_A$,

$$(p \cong_A q) \Rightarrow ((p/Pre_A(p)) \sim_B (q/Pre_A(p)))$$

Of course we get our most intuitive result again at initial states, where Pre sets are void.

*The upward dashed arrows represent Pre transitions

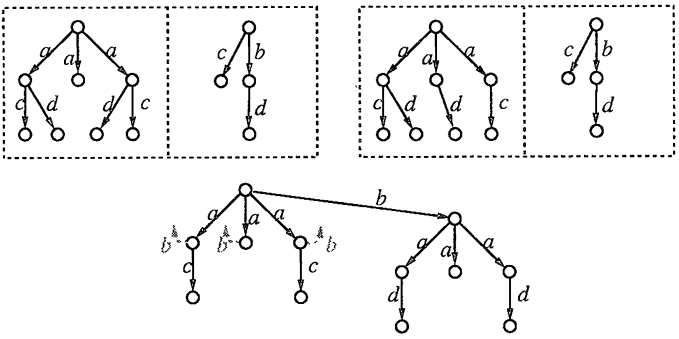

Figure 10 **P**-bisimilar, non bisimilar nets with deterministic Pre actions

4.4 Weak behaviours and τ-reductions

Realistic process descriptions often include hidden local behaviours, and corresponding weak or branching bisimulation equivalences. Defining such notions on partial automata is still a topic of ongoing research. Due to the fact that different local behaviours, sharing the τ label, may recover different independence characteristics, it seems impossible to delay only some of them without getting in a confused case.

A notable exception is when a process p may perform a single τ-action while delaying all other behaviours. In this case, the τ behaviour, independent with all others possible out of p, can safely be omitted. Let's suppose $p \xrightarrow{\tau} q$, and that all other transitions from p can be delayed. In this case, we can collapse p and q in a way that respects branching and weak bisimulations of the complete forms. This kind of reduction was also independently noticed by Gerth et al in (Gerth and Kuiper and Peled and Penczek 1985).

5 PRACTICAL APPLICATIONS

We have implemented these ideas in a prototype system that reduces networks of processes and apply on-the-fly τ-reductions as described above to the partial product. The algorithms, not described here, implement some of the ideas proposed by (Wolper and Godefroid 1993) and (Peled 1993). We are also developing on-the-fly comparison, with partial deterministic specifications for the time being. Our prototype is interfaced to the verification system Auto (de Simone and Vergamini 1989) through the FC2 format (Bouali and Lara de Souza and Madelaine and Ressouche and Roy and de Simone 1995), an exchange representation format for automata and networks. Some of the tests that have been done with the classical example of the Milner's scheduler (described in (Milner 1989)) are described in table 1. We compare it with the global automata and its reduction by weak bisimulation. In this example, we can see that the progression of the number of states of the partial automaton is linear with respect to the number of cyclers of the scheduler, whereas the number of states of the global product grows exponentially.

Table 1 Reductions for Milner's scheduler in the number of states.

cyclers	global product	minimal weak	partial product	partial with τ reductions
3	49	24	10	6
5	321	160	16	10
8	4,097	2,048	25	16
10	20,480	10,240	31	20

Table 2 Reductions for the "Dining Philosophers" problem.

philos.	global product	minimal weak	partial product	partial with τ reductions
2	34	13	16	10
3	214	75	49	32
4	1,294	53	121	80
5	7,774		271	183
6	46,654		577	397
7	279,934		1,195	834

Table 2 describes the results obtained for the well known "Dining Philosophers" problem. The progression in the number of states is exponential for both the partial and the global products. However, the gain in size is important. More precisely, the total number of states for the global product is equal to $6^n - 2$, whereas for the partial product it can be well approximated by $n^{2.295}(1.454)^n$ (where n is the number of philosophers).

6 CONCLUSION

We described a method for comparing two distributed descriptions by constructing only partial reductions of their underlying global systems, which under determinacy assumption provides potential large savings over state spaces. We precisely identified generic counterexamples to the applicability of the method in more general settings. Our restrictions can be seen as imposing conditions on "partialization" functions, according to which behavioural equivalence is to be retained from complete to incomplete automata.

The distributed description formalism we used is still restrictive (no hiding abstraction, no auto-concurrency, and more generally: fixed location set for each action name). While some of these restrictions are inherent to the approach, one can attempt to relax others to some extent.

REFERENCES

Bednarczyk, M.A. (1988) Categories of asynchronous systems. *U. of Sussex*, **1/88**.

Bouali, A. and Lara de Souza, M. and Madelaine, E. and Ressouche, A. and Roy, V. and de Simone, R. (1995) FC2 transformations for clever verification. *Technical report, INRIA.*

de Simone, R. and Vergamini, D. (1989) Aboard Auto. *Technical report, INRIA.*

Mazurkiewicz, A. (1986) Trace Theory. *Petri Nets: Applications and Relationships to Other Models of Concurrency, Advances in Petri Nets 1986, Part II; Proceedings of an Advanced Course.*

Milner, R.. (1989) Communication and Concurrency. *Prentice Hall, Englewood Cliffs.*

Peled, D. (1993) All from One, One for All: On Model Checking Using Representatives. *Proceedings of CAV'93, LNCS 697.*

Peled, D. (1994) Combining Partial Order Reductions with On-the-fly Model-Checking. *Proceedings of CAV'94, LNCS 818.*

Gerth, R. and Kuiper, R. and Peled, R. and Penczek, W. (1995) A Partial Order Approach to Branching Time Model Checking. *Proceedings of ISTCS*, 330–339.

Shields, M.W. (1985) Concurrent machines. *Computer Journal*, **28**.

Valmari, A. (1990) A Stubborn Attack on State Explosion. *Lecture Notes in Computer Science, Springer-Verlag*, **531**.

Winskel, G. and Nielsen, M. (1993) Models for Concurrency. *TEMPUS Summer School on Algebraic and Categorical Methods in Computer Science.*

Wolper, P. and Godefroid, P. (1993) Partial-Order Methods for Temporal Verification. *Proceedings of Concur'93*, **715**.