Minimization Algorithm for Symbolic Bisimilarity*

Filippo Bonchi[1,2] and Ugo Montanari[1]

[1] Dipartimento di Informatica, Università di Pisa
[2] Centrum voor Wiskunde en Informatica (CWI)

Abstract. The operational semantics of interactive systems is usually described by labeled transition systems. Abstract semantics is defined in terms of bisimilarity that, in the finite case, can be computed via the well-known *partition refinement algorithm*. However, the behaviour of interactive systems is in many cases infinite and thus checking bisimilarity in this way is unfeasible. *Symbolic semantics* allows to define smaller, possibly finite, transition systems, by employing symbolic actions and avoiding some sources of infiniteness. Unfortunately, the standard partition refinement algorithm does not work with symbolic bisimilarity.

1 Introduction

The operational semantics of interactive system is usually specified by labeled transition systems (LTSs). Behavioural equivalence is often defined as bisimilarity, namely the largest bisimulation. Many efficient algorithms and tools for bisimulation checking in the finite case have been developed [21,7,8]. Among these, the *partition refinement algorithm* [11,18] is the best known: first it generates the state space of the LTS (i.e., the set of reachable states); then, it creates a partition equating all the states and then, iteratively, refines this partitions by splitting non equivalent states. At the end, the resulting partition equates all and only the bisimilar states.

Most importantly, the same algorithm can be used to construct the *minimal automaton*, that is the smallest (in terms of states and transitions) LTS amongst all those bisimilar. Construction of minimal automata allows to model check efficiently for several properties by eliminating redundant states once and for all. In fact most model checking logics are *adequate w.r.t. bisimilarity*, namely a formula holds in the given system iff it holds in its minimal representative.

In practical cases, *compositionality* is also very relevant, since it is the key to master complexity. Then a fundamental property is that bisimilarity be a congruence. When this is not the case, behavioural equivalence is defined either as the *largest congruence contained into bisimilarity* [13] or as the *largest bisimulation that is also a congruence* [17]. In this paper we focus on the latter and we call it *saturated bisimilarity*. Indeed it coincides with ordinary bisimilarity on

* This work was carried out during the tenure of an ERCIM "Alain Bensoussan" Fellowship Programme and supported by the IST 2004-16004 SENSORIA.

G. Castagna (Ed.): ESOP 2009, LNCS 5502, pp. 267–284, 2009.

the *saturated transition system*, that is obtained by the original LTS by adding the transition $p \xrightarrow{c,a} q$, for every context c, whenever $c(p) \xrightarrow{a} q$.

Many interesting abstract semantics are defined in this way. For example, since late and early bisimilarity of π-calculus [14] are not preserved under substitution (and thus under input prefixes), in [20] Sangiorgi introduces *open bisimilarity* as the largest bisimulation on π-calculus agents which is closed under substitutions. Other noteworthy examples are asynchronous π-calculus [1,10] and mobile ambients calculus [6,12]. The definition of saturated bisimilarity as ordinary bisimulation on the saturated LTS, while in principle operational, often makes infinite state the portion of LTS reachable by any nontrivial agent, and in any case is very inefficient, since it introduces a large number of additional states and transitions. Inspired by [9], Sangiorgi defines in [20] a symbolic transition system and symbolic bisimilarity that efficiently characterizes open bisimilarity. After this, many formalisms have been equipped with a symbolic semantics.

In [4], we have introduced a general model that describes at an abstract level both saturated and symbolic semantics. In this abstract setting, a symbolic transition $p \xrightarrow{c,\alpha}_{\beta} p'$ means that $c(p) \xrightarrow{\alpha} p'$ and c is a smallest context that allows p to performs such transition. Moreover, a certain *derivation relation* $\vdash$ amongst the transitions of a systems is defined: $p \xrightarrow{c_1,\alpha_1} p_1 \vdash p \xrightarrow{c_2,\alpha_2} p_2$ means that the latter transition is a logical consequence of the former. In this way, if all and only the saturated transitions are logical consequences of symbolic transitions, then saturated bisimilarity can be retrievied via the symbolic LTS.

However, the ordinary bisimilarity over the symbolic transition system differs from saturated bisimilarity. Symbolic bisimilarity is thus defined with an asymmetric shape. In the bisimulation game, when a player proposes a transition, the opponent can answer with a move with a different label. For example in the open π-calculus, a transition $p \xrightarrow{[a=b],\tau} p'$ can be matched by $q \xrightarrow{\tau} q'$. Moreover, the bisimulation game does not restart from p' and q', but from p' and $q'\{b/a\}$.

For this reason, algorithms and tools developed for bisimilarity cannot be reused for symbolic bisimilarity. Inspired by [19,15] who developed ad hoc partition refinement algorithm for open and asynchronous bisimilarity, in this paper we introduce a generical *symbolic partition refinement algorithm*, relying on the theoretical framework presented in [4]. The algorithm is based on the notion of *redundant symbolic transitions*. Intuitively, a symbolic transition $p \xrightarrow{c_2,\alpha_2}_{\beta} q$ is redundant if there exists another symbolic transition $p \xrightarrow{c_1,\alpha_1}_{\beta} p_1$ that logically implies it, that is $p \xrightarrow{c_1,\alpha_1}_{\beta} p_1 \vdash p \xrightarrow{c_2,\alpha_2} p_2$ and q is bisimilar to p_2. Now, if we consider the LTS having only not-redundant transitions, the ordinary notion of bisimilarity coincides with saturated bisimilarity. Thus, in principle, we could remove all the redundant transitions and then check bisimilarity with the standard partition refinement algorithm. But, how can we decide which transitions are redundant, if redundancy itself depends on bisimilarity?

Our solution consists in computing bisimilarity and redundancy *at the same time*. In the first step, the algorithm considers all the states bisimilar and all the transitions (that are potentially redundant) as redundant. At any iteration,

states are distinguished according to (the current estimation of) not-redundant transitions and then not-redundant transitions are updated according to the new computed partition. The main peculiarity of the algorithm is that in the initial partition, we have to insert not only the reachable states, but also those that are needed to check redundancy. An extended version of the paper is in [5].

2 Partition Refinement and Minimal Automaton

In CCS [13], bisimilarity ($\sim$) is defined as the largest bisimulation relation, i.e., the largest relation R such that $R \subseteq \mathbf{F}(R)$ where $\mathbf{F}$ is a function such that for each relation R, $p\,\mathbf{F}(R)\,q$ iff

- if $p \xrightarrow{a} p'$ then $q \xrightarrow{a} q'$ and $p' R q'$,
- if $q \xrightarrow{a} q'$ then $p \xrightarrow{a} p'$ and $p' R q'$.

Since $\mathbf{F}$ is monotonic for set inclusion, $\sim = \bigcup\{R \mid R \subseteq \mathbf{F}(R)\}$ follows from standard results on fixed point theory. Moreover, $\sim$ is itself a fix point of $\mathbf{F}$, i.e., $\sim = \mathbf{F}(\sim)$. Alternatively, bisimilarity can be characterized as the limit of a decreasing chains of relations (none of them is a bisimulation) starting with the universal relation. Hereafter, we use κ to denote ordinals numbers, $\kappa+1$ for successor of κ, λ for limits ordinals and $\mathcal{O}$ for the class of all ordinals. Formally, the *terminal sequence* is defined for each ordinal κ as follow,

$$\sim^0 = \{\mathcal{P} \times \mathcal{P}\} \quad \sim^{\kappa+1} = \mathbf{F}(\sim^\kappa) \quad \sim^\lambda = \bigcap_{\kappa<\lambda} \sim^\kappa$$

where $\mathcal{P}$ is the set of all CCS processes.

Bisimilarity coincides with the limit of the terminal sequence.

Proposition 1. $\sim = \bigcap_{\kappa\in\mathcal{O}} \sim^\kappa$

Given a set S, a *partition* of S is a set of *blocks*, i.e. subsets of S, that are all disjoint and whose union is S. A partition on S represents an equivalence relation, where equivalent elements belong to the same block. In the following, given a function $\mathbf{G}$ on equivalence relations, we denote by $\overline{\mathbf{G}}$ the corresponding function on partitions.

The characterization of bisimilarity through the terminal sequence suggests a procedure for checking bisimilarity of a set of initial states IS. First of all, we compute $IS^\star$, i.e., the set of all states that are reachable from IS. Then we create the partition P^0 where all the elements of $IS^\star$ belongs to the same block. After the initialization, we iteratively refine the partitions by using the function $\overline{\mathbf{F}}$ (i.e., the function equivalent to $\mathbf{F}$ on partitions): two states p and q belong to the same block in P^{n+1}, if and only if whenever $p \xrightarrow{a} p'$ then $q \xrightarrow{a} q'$ with p' and q' in the same block of P^n and viceversa.The algorithm terminates whenever two consecutive partitions are equivalent. In such partition two states belong to the same block if and only if they are bisimilar. Notice that since $\mathbf{F}$ is monotonic, any iteration splits blocks and never fuse them. For this reason if $IS^\star$ is finite, the algorithm terminates in at most $|IS^\star|$ iterations.

Algorithm 1. `Partition-Refinement(`IS`)`

Initialization

1. $IS^\star$ is the set of all processes reachable from IS,
2. $P^0 := \{IS^\star\}$,

Iteration $P^{n+1} := \overline{\mathbf{F}}(P^n)$,
Termination If $P^n = P^{n+1}$ then return P^n.

Proposition 2. *If $IS^\star$ is finite, then the algorithm terminates and the resulting partition equates all and only the bisimilar state.*

The partition refinement algorithm allows not only to check bisimilarity of a set of states, but also to build the *minimal automaton* of a certain state p. Intuitively, the minimal automaton is a labeled transition systems where all the bisimilar states are identified. Hereafter, given a set A and an equivalence relation R, we write $A_{|R}$ to denote the set of equivalence classes of A w.r.t. R. Moreover, given $p \in A$, $[p]_R$ denotes the equivalence class of p w.r.t. R.

Definition 1 (Minimal Automaton). *Let $\{p\}^\star$ be the set of states reachable from the state p. The* minimal automaton of p *(denoted by $MA(p)$) is a triple $\langle i, M, tr_M \rangle$:*

- *the initial state i is equal to $[p]_\sim$,*
- *$M = \{p\}^\star_{|\sim}$ is the set of equivalence classes of $\sim$,*
- *tr_M is the transition relation defined according to the following rule.*

$$\frac{q \xrightarrow{a} r}{[q]_\sim \xrightarrow{a}_M [r]_\sim}$$

Proposition 3. *$p \sim q$ if and only if $MA(p)$ is isomorphic to $MA(q)$.*

If the set of states reachable from p is finite, we can employ the partition refinement algorithm to build the minimal automaton of p. We have just to quotient the set of reachable states $\{p\}^\star$ with the partition returned by the `Partition-Refinement(`$\{p\}$`)`.

3 Saturated and Symbolic Semantics

In this section we recall the general framework for symbolic bisimilarity that we have introduced in [4]. As running example, we will use open Petri nets [2]. However, our theory has as special cases the abstract semantics of several formalisms such as open [20] and asynchronous [1] π-calculus.

3.1 Saturated Semantics

A *closed many-sorted unary signature* (S, Σ) consists of a set of sorts S, and an $S \times S$ sorted family $\Sigma = \{\Sigma_{s,t} \mid s, t \in S\}$ of sets of operation symbols which are

closed under composition, that is if $f \in \Sigma_{s,t}$ and $g \in \Sigma_{t,u}$, then $g \circ f \in \Sigma_{s,u}$. Given $f \in \Sigma_{u,v}, g \in \Sigma_{t,u}, h \in \Sigma_{s,t}$, $f \circ (g \circ h) = (f \circ g) \circ h$ and moreover $\forall s \in S$, $\exists id_s \in \Sigma_{s,s}$ such that $\forall f \in \Sigma_{s,t}$, $id_t \circ f = f$ and $f \circ id_s = f$. A (S, Σ)-algebra $\mathbb{A}$ consists of an S sorted family $|\mathbb{A}| = \{A_s \mid s \in S\}$ of sets and a function $f_{\mathbb{A}} : A_s \to A_t$ for all $f \in \Sigma_{s,t}$ such that $(g \circ f)_{\mathbb{A}} = g_{\mathbb{A}}(f_{\mathbb{A}}(-))$ and $id_{s\mathbb{A}}$ is the identity function on A_s[1]. When $\mathbb{A}$ is clear from the context, we will write f to mean $f_{\mathbb{A}}$, and we will write A_s to mean the set of sort s of the family $|\mathbb{A}|$.

The first definition of the theoretical framework presented in [4] is that of *context interactive systems*. In our theory, an interactive system is a state-machine that can interact with the environment (contexts) through an evolving interface.

Definition 2 (Context Interactive System). *A* context interactive system *$\mathcal{I}$ is a quadruple $\langle (S, \Sigma), \mathbb{A}, O, tr \rangle$ where:*

- *(S, Σ) is a closed many-sorted unary signature,*
- *$\mathbb{A}$ is a (S, Σ)-algebra,*
- *O is a set of observations,*
- *$tr \subseteq |\mathbb{A}| \times O \times |\mathbb{A}|$ is a labeled transition relation ($p \xrightarrow{o} p'$ means $(p, o, p') \in tr$).*

Roughly speaking sorts are interfaces of the system, while operators of Σ are contexts. Every state p with interface s (i.e. $p \in A_s$) can be inserted into the context $c \in \Sigma_{s,t}$, obtaining $c_{\mathbb{A}}(p)$ that has interface t. Every state can evolve into a new state (possibly with different interface) producing an observation $o \in O$.

The abstract semantics of interactive systems is usually defined through behavioural equivalences. In [4] we proposed a general notion of bisimilarity that generalizes the abstract semantics of a large variety of formalisms. The idea is that two states of a system are equivalent if they are indistinguishable from an external observer that, in any moment of their execution, can insert them into some environment and then observe some transitions.

Definition 3 (Saturated Bisimilarity). *Let $\mathcal{I} = \langle (S, \Sigma), \mathbb{A}, O, tr \rangle$ be a context interactive system. Let $R = \{R_s \subseteq A_s \times A_s \mid s \in S\}$ be an S sorted family of symmetric relations. R is a* saturated bisimulation *iff, $\forall s, t \in S$, $\forall c \in \Sigma_{s,t}$, whenever pR_sq:*

- *$c_{\mathbb{A}}(p)\, R\, c_{\mathbb{A}}(q)$,*
- *if $p \xrightarrow{o} p'$, then $q \xrightarrow{o} q'$ and $p' R q'$.*

We write $p \sim_s^S q$ iff there is a saturated bisimulation R such that pR_sq.

An alternative but equivalent definition can be given by defining the *saturated transition system* (SATTS) as follows: $p \xrightarrow{c,o}_S q$ if and only if $c(p) \xrightarrow{o} q$. Trivially the ordinary bisimilarity over SATTS coincides with $\sim^S$.

Proposition 4. *$\sim^S$ is the coarsest bisimulation congruence.*

[1] A closed many-sorted unary signature (S, Σ) is a category $\mathbf{C}$ and a (S, Σ)-algebra is a presheaf on $\mathbf{C}$. We adopt the above notation to be accessible to a wider audience.

3.2 Running Example: Open Petri Nets

Differently from process calculi, Petri nets have not a widely known interactive behaviour. Indeed they model concurrent systems that are closed, in the sense that they do not interact with the environment. *Open nets* [2] are P/T Petri nets that can interact by exchanging tokens on *input* and *output places*.

Definition 4 (Open Net). *An* open net *is a tuple* $N = (S, T, pre, post, l, I, O)$ *where* S *and* T *are the sets of places and transitions* ($S \cap T = \varnothing$)*;* $pre, post : T \to S^{\oplus}$ *are functions mapping each transition to its pre- and post-set;* $l : T \to \Lambda$ *is a labeling function (*Λ *is a set of labels) and* $I, O \subseteq S$ *are the sets of input and output places. A* marked open net *is a pair* $\langle N, m \rangle$ *where* N *is an open net and* $m \in S^{\oplus}$ *is a marking.*

Fig.1 shows five open nets where, as usual, circles represents places and rectangles transitions (labeled with α, β). Arrows from places to transitions represent *pre*, while arrows from transitions to places represent *post*. Input places are denoted by ingoing edges, thus the only input place of N_1 is \$. To make examples easier, hereafter we only consider *open input nets*, i.e., open nets without output places. The operational semantics of marked open nets is expressed by the rules on Table 1.The rule (TR) is the standard rule of P/T nets (seen as multisets rewriting). The rule (IN) states that in any moment a token can be inserted inside an input place and, for this reason, the LTS has always an infinite number of states. Fig.1(A) shows part of the infinite transition system of $\langle N_1, a \rangle$. The abstract semantics (denoted by $\sim^N$) is defined in [3] as the ordinary bisimilarity over such an LTS. It is worth noting that $\sim^N$ can be seen as an instance of saturated semantics, where multisets over open places are contexts and transitions are only those generated by the rule (TR).

In the following we formally define $\mathcal{N} = \langle (S^{\mathcal{N}}, \Sigma^{\mathcal{N}}), \mathbb{N}, \Lambda, tr_{\mathcal{N}} \rangle$ that is the context interactive system of all open nets (labeled over the set of labels Λ).

The many-sorted signature $(S^{\mathcal{N}}, \Sigma^{\mathcal{N}})$ is formally defined as:

- $S^{\mathcal{N}} = \{I \mid I \text{ is a set of places}\}$,
- $\forall I \in S^{\mathcal{N}}$, $\Sigma^{\mathcal{N}}_{I,I} = I^{\oplus}$, $id_I = \varnothing$ and $i_1 \circ i_2 = i_1 \oplus i_2$.

Intuitively sorts are sets of input places I, while operators of $\Sigma^{\mathcal{N}}$ are multisets of tokens on the input places. We say that a marked open net $\langle N, m \rangle$ has interface I if the set of input places of N is I. For example the marked open nets $\langle N_1, a \rangle$ has interface $\{\$\}$. Let us define the $(S^{\mathcal{N}}, \Sigma^{\mathcal{N}})$-algebra $\mathbb{N}$. For any sort I, the carrier set N_I contains all the marked open nets with interface I. Any operator $i \in \Sigma_{I,I}$ is defined as the function that maps $\langle N, m \rangle$ into $\langle N, m \oplus i \rangle$.

The transition structure $tr_{\mathcal{N}}$ (denoted by $\rightarrow_{\mathcal{N}}$) associates to a state $\langle N, m \rangle$ the transitions obtained by using the rule (TR) of Table 1. In [4], it is proved that

Table 1. Operational Semantics of marked open nets

$$(\text{TR})\ \frac{t \in T \quad l(t) = \lambda \quad m = {}^{\bullet}t \oplus c}{N, m \xrightarrow{\lambda} N, t^{\bullet} \oplus c} \qquad (\text{IN})\ \frac{i \in I_N}{N, m \xrightarrow{+i} N, m \oplus i}$$

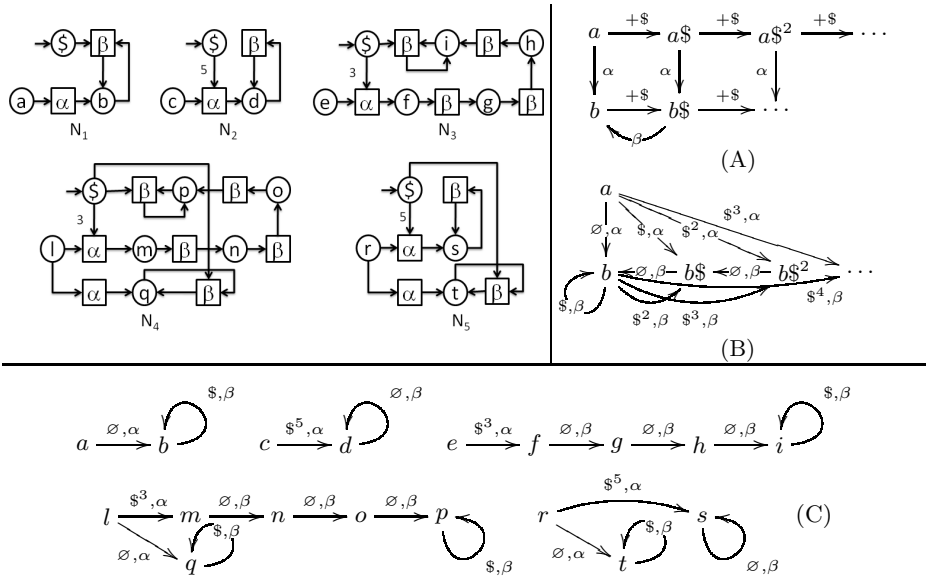


Fig. 1. The open nets N_1, N_2, N_3, N_4 and N_5.(A)Part of the infinite transition system of $\langle N_1, a\rangle$. (B)Part of the infinite saturated transition system of $\langle N_1, a\rangle$.(C)The symbolic transition systems of $\langle N_1, a\rangle$,$\langle N_2, c\rangle$,$\langle N_3, e\rangle$,$\langle N_4, l\rangle$ and $\langle N_5, r\rangle$.

saturated bisimilarity for $\mathcal{N}$ coincides with $\sim^N$. In the remainder of the paper we will use as running example the open nets in Fig.1. Since all the places have different names (with the exception of \$), in order to make lighter the notation, we write only the marking to mean the corresponding marked net, e.g. $b^2\$$ means the marked net $\langle N_1, b^2\$\rangle$.

The marked net a (i.e., $\langle N_1, a\rangle$) represents a system that provides a service β. After the activation α, it provides β whenever the client pay one \$ (i.e., the environment insert a token into \$). The marked net c instead requires five \$ during the activation, but then provides the service β for free. The marked net e, requires three \$ during the activation. For three times, the service β is performed for free and then it costs one \$. It is easy to see that all these marked nets are not bisimilar. Indeed, a client that has only one \$ can have the service β only with a, while a client with five \$ can have the service β for six times only with c. The marked net r represents a system that offers the behaviour of both a and c, i.e. either the activation α is for free and then the service β costs one, or the activation costs five and then the service is for free. Also this marked net is different from all the others.

Now consider the marked net l. It offers the behaviour of both a and e, but it is equivalent to a, i.e. $l \sim^N a$. Roughly, the behaviour of e is absorbed by the behaviour of a. This is analogous to what happens in asynchronous π-calculus [1] where it holds that $a(x).(\overline{a}x \mid p) + \tau.p \sim \tau.p$.

3.3 Symbolic Semantics

Saturated bisimulation is a good notion of equivalence but it is hard to check, since it involves a quantification over all contexts. In [4], we have introduced a general notion of *symbolic bisimilarity* that coincides with saturated bisimilarity, but it avoids to consider all contexts. The idea is to define a symbolic transition system where transitions are labeled both with the usual observation and also with the minimal context that allows the transition.

Definition 5 (Symbolic Context Transition System). *A* symbolic context transition system *(*SCTS *for short) for a system* $\mathcal{I} = \langle (S, \Sigma), \mathbb{A}, O, tr \rangle$ *is a transition system* $\beta \subseteq |\mathbb{A}| \times \Sigma \times O \times |\mathbb{A}|$.

In [4], we have introduced a SCTS for open nets. Intuitively the symbolic transition $N, m \xrightarrow{i,\lambda}_{\eta} N, m'$ is possible if and only if $N, m \oplus i \xrightarrow{\lambda}_{\mathcal{N}} N, m'$ and i is the smallest multiset (on input places) allowing such transition. This SCTS is formally defined by the following rule.

$$\frac{t \in T \quad l(t) = \lambda \quad m = (m \cap {}^\bullet t) \oplus n \quad i \subseteq I^{\oplus} \quad {}^\bullet t = (m \cap {}^\bullet t) \oplus i}{N, m \xrightarrow{i,\lambda}_{\eta} N, t^\bullet \oplus n}$$

The marking $m \cap {}^\bullet t$ contains all the tokens of m that are needed to perform the transition t. The marking n contains all the tokens of m that are not useful for performing t, while the marking i contains all the tokens that m needs to reach ${}^\bullet t$. Note that i is exactly the *smallest* multiset that is needed to perform the transition t. Indeed if we take i_1 strictly included into i, $m \oplus i_1$ cannot match ${}^\bullet t$. As an example consider the net N_2 in Fig.1 with marking $cd\$^2$ and let t be the only transition labeled with α. We have that $cd\$^2 \cap {}^\bullet t = c\$^2$, $n = d$ and $i = \$^3$. Thus $N_2, cd\$^2 \xrightarrow{\$^3,\alpha}_{\eta} N_2, dd$. Fig.1(C) shows symbolic transition systems of marked open nets discussed in the previous subsection.

Definition 6 (Inference System). *An inference system* $\mathcal{R}$ *for a context interactive system* $\mathcal{I} = \langle (S, \Sigma), \mathbb{A}, O, tr \rangle$ *is a set of rules of the following format, where* $s, t \in S$, $o, o' \in O$, $c \in \Sigma_{s,s'}$ *and* $d \in \Sigma_{t,t'}$.

$$\frac{p_s \xrightarrow{o} q_t}{c(p_s) \xrightarrow{o'} d(q_t)}$$

The above rule states that all processes with sort s that perform a transition with observation o going into a state q_t with sort t, when inserted into the context c can perform a transition with the observation o' going into $d(q_t)$.

In the following, we write $c \xrightarrow[o']{o} d$ to mean a rule like the above. The rules $c \xrightarrow[o']{o} c'$ and $d \xrightarrow[o'']{o'} d'$ *derive* the rule $d \circ c \xrightarrow[o'']{o} d' \circ c'$ if $d \circ c$ and $d' \circ c'$ are defined. Given an inference system $\mathcal{R}$, $\Phi(\mathcal{R})$ is the set of all the rules derivable from $\mathcal{R}$ together with the identities rules ($\forall o \in O$ and $\forall s, t \in S$, $id_s \xrightarrow[o]{o} id_t$).

Definition 7 (Derivations, soundness and completeness). *Let $\mathcal{I}$ be a context interactive system, β an* SCTS *and $\mathcal{R}$ an inference system.*

We say that $p \xrightarrow{c_1,o_1} p_1$ derives *$p \xrightarrow{c_2,o_2} p_2$ in $\mathcal{R}$ (written $p \xrightarrow{c_1,o_1} p_1 \vdash_{\mathcal{R}} p \xrightarrow{c_2,o_2} p_2$) if there exist $d, e \in \Sigma$ such that $d \ \frac{o_1}{o_2}\ e \in \Phi(\mathcal{R})$, $d \circ c_1 = c_2$ and $e_{\mathbb{A}}(p_1) = p_2$.*

We say that β and $\mathcal{R}$ are sound and complete *w.r.t. $\mathcal{I}$ if*

$$p \xrightarrow{c,o}_S q \text{ iff } p \xrightarrow{c',o'}_\beta q' \text{ and } p \xrightarrow{c',o'}_\beta q' \vdash_{\mathcal{R}} p \xrightarrow{c,o}_S q.$$

A sound and complete SCTS could be considerably smaller than the saturated transition system, but still containing all the information needed to recover $\sim^S$. Note that the ordinary bisimilarity over SCTS (hereafter called *syntactical bisimilarity* and denoted by $\sim^W$) is usually stricter than $\sim^S$. As an example consider the marked open nets a and l. These are not syntactically bisimilar, since $l \xrightarrow{\$^3,\alpha}_\eta m$ while a cannot (Fig.1(C)). However, they are saturated bisimilar, as discussed in the previous subsection. In order to recover $\sim^S$ through the symbolic transition system we need a more elaborated definition of bisimulation.

Definition 8 (Symbolic Bisimilarity). *Let $\mathcal{I} = \langle (S, \Sigma), \mathbb{A}, O, tr \rangle$ be an interactive system, $\mathcal{R}$ be a set of rules and β be a symbolic transition system. Let $R = \{R_s \subseteq A_s \times A_s \mid s \in S\}$ be an S sorted family of symmetric relations. R is a* symbolic bisimulation *iff $\forall s \in S$, whenever pR_sq:*

- *if $p \xrightarrow{c,o}_\beta p'$, then $q \xrightarrow{c_1,o_1}_\beta q_1'$ and $q \xrightarrow{c_1,o_1}_\beta q_1' \vdash_{\mathcal{R}} q \xrightarrow{c,o} q'$ and $p'Rq'$.*

We write $p \sim_s^{SYM} q$ iff there exists a symbolic bisimulation R such that pR_sq.

Theorem 1. *Let $\mathcal{I}$ be a context interactive system, β an* SCTS *and $\mathcal{R}$ an inference system. If β and $\mathcal{R}$ are sound and complete w.r.t. $\mathcal{I}$, then $\sim^{SYM} = \sim^S$.*

In the remainder of this section we focus on open Petri nets. The inference system $\mathcal{R}_\mathcal{N}$ is defined by the following parametric rule.

$$\frac{N, m \xrightarrow{\lambda}_\mathcal{N} N, m'}{N, m \oplus i \xrightarrow{\lambda}_\mathcal{N} N, m' \oplus i}$$

The intuitive meaning of this rule is that for all possible observations λ and multiset i on input places, if a marked net performs a transition with observation λ, then the addition of i preserves this transition.

Now, consider derivations between transitions of open nets. It is easy to see that $N, m \xrightarrow{i_1,\lambda_1} N, m_1 \vdash_{\mathcal{R}_\mathcal{N}} N, m \xrightarrow{i_2,\lambda_2} N, m_2$ if and only if $\lambda_2 = \lambda_1$ and there exists a multiset x on input places such that $i_2 = i_1 \oplus x$ and $m_2 = m_1 \oplus x$. For all the nets N_k of our example, this just means that for all observations λ and for all multisets m, n, we have that $\langle N_k, m \rangle \xrightarrow{\$^i,\lambda}_\eta \langle N_k, n \rangle \vdash_{\mathcal{R}_\mathcal{N}} \langle N_k, m \rangle \xrightarrow{\$^{i+j},\lambda} \langle N_k, n\$^j \rangle$.

In [4] we have shown that $\mathcal{R}_\mathcal{N}$ and η are sound and complete w.r.t. $\mathcal{N}$. For this reason, we can prove that two marked nets are bisimilar, by showing a symbolic bisimulation that relates them.

4 Saturated Terminal Sequences

In this section we introduce the terminal sequence for saturated and symbolic bisimilarity. They are almost straightforward adaptation of the terminal sequence for ordinary bisimilarity presented in Section 2. Hereafter we always implicitly refer to a context interactive system $\mathcal{I} = \langle (S, \Sigma), \mathbb{A}, O, tr \rangle$, a SCTS β and an inference system $\mathcal{R}$, such that β and $\mathcal{R}$ are sound and complete w.r.t. $\mathcal{I}$.

The *saturated terminal sequence* is defined as follows,

$$\sim_S^0 = \{A_s \times A_s \mid s \in S\} \quad \sim_S^{\kappa+1} = \mathbf{SAT}(\sim_S^\kappa) \quad \sim_S^\lambda = \bigcap_{\kappa<\lambda} \sim_S^\kappa$$

where **SAT** is a function on S indexed families of relations such that, for all $R = \{R_s \subseteq A_s \times A_s \mid s \in S\}$, $p\mathbf{SAT}(R)q$ iff

- if $p \xrightarrow{c,o}_S p'$, then $q \xrightarrow{c,o}_S q'$ and $p'Rq'$,
- if $q \xrightarrow{c,o}_S q'$, then $p \xrightarrow{c,o}_S p'$ and $p'Rq'$.

The only difference w.r.t. the terminal sequence of ordinary bisimilarity is in the fact that we consider S indexed families of relations (recall that S is the set of sorts, and A_s is carrier set of sort s of the algebra $\mathbb{A}$).

It is easy to see that **SAT** is monotonic w.r.t. (indexed) set inclusion. From classical results of fixed point theory (analogously to ordinary bisimilarity), we have that saturated bisimilarity is the limit of the saturated terminal sequence.

Proposition 5. $\sim^S = \bigcap_{\kappa \in \mathcal{O}} \sim_S^\kappa$

The following lemma is fundamental to prove the correctness of our algorithm.

Lemma 1. *$\forall \kappa \in \mathcal{O}$, $\sim_S^\kappa$ is a congruence.*

In Section 2, we have shown that the terminal sequence for ordinary bisimilarity provides an effective procedure for computing bisimilarity. We would like to apply the same intuition to the saturated terminal sequence but, unfortunately, the saturated transition system is usually infinite, since it considers all possible contexts. Instead of using the SATTS, we could define the *symbolic terminal sequence* relying just on the symbolic transition system. However, also this approach immediately leads to work with infinitely many states.

5 Redundant Transitions

In Section 3, we have shown that syntactical bisimilarity ($\sim^W$), i.e. the ordinary bisimilarity on the symbolic transition system, does not coincide with $\sim^S$. Here we show that this is due to the presence of *redundant transitions*. In order to better explain this phenomenon, we have to show an important property of $\vdash_\mathcal{R}$.

Lemma 2. *$\forall p, q$, if $p \xrightarrow{c_1,d_1} p_1 \vdash_\mathcal{R} p \xrightarrow{c_2,d_2} e_\mathbb{A}(p_1)$, then $q \xrightarrow{c_1,d_1} q_1 \vdash_\mathcal{R} q \xrightarrow{c_2,d_2} e_\mathbb{A}(q_1)$.*

Now, consider a process p that performs only the symbolic transitions $p \xrightarrow{c_1,o_1}_\beta p_1$ and $p \xrightarrow{c_2,o_2}_\beta p_2$ such that $p \xrightarrow{c_1,o_1}_\beta p_1 \vdash_\mathcal{R} p \xrightarrow{c_2,o_2} e_\mathbb{A}(p_1)$ and $p_2 \sim^S e_\mathbb{A}(p_1)$. The transition $p \xrightarrow{c_2,o_2}_\beta p_2$ is *redundant* and it makes $\sim^W$ different from $\sim^S$. Indeed, take a process q that performs only $q \xrightarrow{c_1,o_1}_\beta q_1$ such that $p_1 \sim^S q_1$. Clearly p and q are not syntactically bisimilar, because $p \xrightarrow{c_2,o_2}_\beta p_2$ while q cannot. However, $p \sim^S q$, because $q \xrightarrow{c_2,o_2}_S e_\mathbb{A}(q_1)$ (assuming that β and $\mathcal{R}$ are sound and complete and by Lemma 2) and, $p_2 \sim^S e_\mathbb{A}(p_1) \sim^S e_\mathbb{A}(q_1)$ (since $\sim^S$ is a congruence).

As an example consider the symbolic transition system of l (Fig.1). $l \xrightarrow{\varnothing,\alpha}_\eta q$ and $l \xrightarrow{\$^3,\alpha}_\eta m$. Moreover, $l \xrightarrow{\varnothing,\alpha}_\eta q \vdash_{\mathcal{R}_\mathcal{N}} l \xrightarrow{\$^3,\alpha} q\$^3$ and $q\$^3 \sim^S m$. Now consider a. $a \xrightarrow{\varnothing,\alpha}_\eta b$. Clearly $l \not\sim^W a$ but they are saturated bisimilar (Section 3).

Definition 9 (Redundant Transition). *Let $\mathcal{I} = \langle (S, \Sigma), \mathbb{A}, O, tr\rangle$ be a context interactive system, $\mathcal{R}$ be an inference system and X be an S sorted family of relations. Let $p \xrightarrow{c_1,o_1} p_1$ and $p \xrightarrow{c_2,o_2} p_2$ be two different transitions. We say that the former* dominates *the latter in X (written $p \xrightarrow{c_1,o_1} p_1 \prec_X p \xrightarrow{c_2,o_2} p_2$) if and only if $p \xrightarrow{c_1,o_1} p_1 \vdash_\mathcal{R} p \xrightarrow{c_2,o_2} e_\mathbb{A}(p_1)$ and $p_2 \, X \, e_\mathbb{A}(p_1)$. A transition is* redundant *w.r.t. X if it is dominated in X by another transition. Otherwise, it is* irredundant.

In the remainder of this section, we introduce another characterization of saturated bisimilarity that only checks irredundant symbolic transitions. The minimization algorithm that we will present in Section 6 relies on this notion.

Definition 10 (Irredundant Bisimilarity). *Let $\mathcal{I} = \langle (S, \Sigma), \mathbb{A}, O, tr\rangle$ be an interactive system, $\mathcal{R}$ be a set of rules and β be a symbolic transition system. Let $R = \{R_s \subseteq A_s \times A_s \mid s \in S\}$ be an S sorted family of symmetric relations. R is an* irredundant bisimulation *iff $\forall s \in S$, whenever pR_sq:*

- *if $p \xrightarrow{c,o}_\beta p'$ is irredundant in R, then $q \xrightarrow{c,o}_\beta q'$ and $p'Rq'$.*

We write $p \sim_s^{NR} q$ iff an irredundant bisimulation R such that pR_sq exists.

Theorem 2 states that $\sim^{NR}$=$\sim^S$. However, in order to have such correspondence, we have to add a constraint to our theory. Indeed, according to the actual definition of context interactive systems, there could exist infinite descending chains like: $\cdots \prec_R p \xrightarrow{c_2,o_2} p_2 \prec_R p \xrightarrow{c_1,o_1} p_1$. In this chain, all the transitions are redundant and thus none of them is considered when checking irredundant bisimilarity.

Definition 11. *A context interactive system is* well-founded *w.r.t. $\mathcal{R}$ if and only if for all relations R there are no infinite descending chains of $\prec_R$.*

All the examples that we have shown in [4] are well-founded. In particular $\mathcal{N}$ is well founded w.r.t. $\mathcal{R}_\mathcal{N}$. Indeed, for all relations R, $m \xrightarrow{i_1,\lambda_1} m_1 \prec_R m \xrightarrow{i_2,\lambda_2} m_2$ only if there exists a multiset $x \neq \varnothing$ such that $x \circ i_1 = i_2$. This means that the multiset i_1 is strictly included in the multiset i_2, and since all multisets are finite, there exist only finite descending chains of $\prec_R$.

Theorem 2. *If $\mathcal{I}$ is well founded w.r.t. $\mathcal{R}$, then $\sim^{NR}$=$\sim^{SYM}$.*

6 A Minimization Algorithm for Symbolic Bisimilarity

In this section we introduce the terminal sequence for irredundant bisimilarity and we prove that it coincides with the saturated terminal sequence (Subsection 6.1). Relying on this, we introduce the symbolic partition refinement algorithm that checks saturated bisimilarity (Subsection 6.2). Finally, we prove the existence of minimal symbolic automata and we provide a procedure to compute them (Subsection 6.3). Hereafter, we assume that $\mathcal{I}$ is well-founded w.r.t. $\mathcal{R}$.

6.1 Irredundant Terminal Sequence

The *irredundant terminal sequence* ($\sim^{\kappa}_{IR}$) is defined as the saturated terminal sequence by replacing the function **SAT** with **IR** that is defined as follows: for all $R = \{R_s \subseteq A_s \times A_s \mid s \in S\}$, $p\mathbf{IR}(R)q$ iff

- if $p \xrightarrow{c,o}_{\beta} p'$ is irredundant in R, then $q \xrightarrow{c,o}_{\beta} q'$ and $p'Rq'$,
- if $q \xrightarrow{c,o}_{\beta} q'$ is irredundant in R, then $p \xrightarrow{c,o}_{\beta} p'$ and $p'Rq'$.

The function **IR** is clearly different from **SAT**, but they are equivalent when restricting to congruences.

Proposition 6. *Let $R = \{R_s \subseteq A_s \times A_s \mid s \in S\}$ be an S sorted family of symmetric relations. If R is a congruence, then $\mathbf{SAT}(R) = \mathbf{IR}(R)$.*

Since by Lemma 1 all the relations of the saturated terminal sequence are congruences, then the two terminal sequences coincide.

Theorem 3. $\forall \kappa \in \mathcal{O}$, $\sim^{\kappa}_{S} = \sim^{\kappa}_{IR}$.

6.2 Symbolic Partition Refinement

In Section 2 we have shown how the terminal sequence can be employed in order to have an effective procedure to compute bisimilarity. In this section we apply the same intuition to the irredundant terminal sequence. At the iteration n, instead of computing $\overline{\mathbf{F}}(P^n)$, we compute $\overline{\mathbf{IR}}(P^n)$: two processes p and q belong to the same block in P^{n+1}, if and only if whenever $p \xrightarrow{c,o}_{\beta} p'$ is not redundant in P^n then $q \xrightarrow{c,o}_{\beta} q'$ with p' and q' in the same block of P^n.

It is worth noting that in the computation of $\overline{\mathbf{IR}}(P^n)$ are involved also states that could be not reachable from the initial states IS. As an example consider the symbolic transition system of a and r (Fig.1(C)). The set of reachable states is $IS^{\star} = \{a, b, r, s, t\}$. Recall that $r \xrightarrow{\varnothing,\alpha}_{\eta} t \vdash_{\mathcal{R}_{\mathcal{N}}} r \xrightarrow{\$^5,\alpha}_{\eta} t\$^5$. Thus, at the generic iteration $n+1$, we need to check if the transition $j \xrightarrow{\$^5,\alpha}_{\eta} s$ is redundant. In order to do that we have to check if $t\$^5$ and s belong to the same block in P^n. However, the state $t\$^5$ is not reachable from $IS = \{a, r\}$.

Thus, we have to change the initialization step of our algorithm, by including in the set $IS^{\star}$ all the states that are needed to check redundancy. This is done,

Table 2. Closure rules

$$(\text{IS})\frac{p \in IS \quad p \in A_s}{p \in IS^\star_s} \qquad (\text{RS})\frac{p \in IS^\star \quad p \xrightarrow{c,o}_\beta q \quad q \in A_s}{q \in IS^\star_s}$$

$$(\text{RD})\frac{p \in IS^\star \quad p \xrightarrow{c_1,o_1}_\beta q_1 \quad p \xrightarrow{c_2,o_2}_\beta q_2 \quad p \xrightarrow{c_1,o_1}_\beta q_1 \vdash_{\mathcal{R}} p \xrightarrow{c_2,o_2} e_{\mathbb{A}}(q_1) \quad e_{\mathbb{A}}(q_1) \in A_s}{e_{\mathbb{A}}(q_1) \in IS^\star_s}$$

by using the closure rules in Table 2.The rule (RD) adds all the states that are needed to check redundancy. Indeed, if p can perform $p \xrightarrow{c_1,o_1}_\beta q_1$ and $p \xrightarrow{c_2,o_2}_\beta q_2$ such that $p \xrightarrow{c_1,o_1}_\beta q_1 \vdash_{\mathcal{R}} p \xrightarrow{c_2,o_2} e_{\mathbb{A}}(q_1)$, the latter could be redundant whenever $q_2 \sim^S e_{\mathbb{A}}(q_1)$. Thus also the state $e_{\mathbb{A}}(q_1)$ is needed. As an example, the closure of $IS = \{a, r\}$ is $IS^\star = \{a, b, r, s, t, t\$^1, t\$^2, t\$^3, t\$^4, t\$^5\}$ (Fig.2(B)). Usually, $IS^\star$ is not just a set, but an S indexed family of sets of states and for this reason the closure rules in Table 2 insert states in $IS^\star$ according to their sorts.

Algorithm 2. `Symbolic-Partition-Refinement(`IS`)`

Initialization

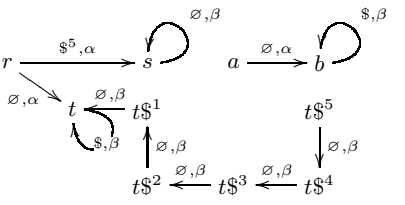
1. Compute $IS^\star$ with the rules in Table 2,
2. $P^0 := \{IS^\star_s | s \in S\}$,

Iteration $P^{n+1} := \overline{\mathbf{IR}}(P^n)$,
Termination if $P^n = P^{n+1}$ then return P^n.

Notice that in the initial partition P^0 there is one block for each sort $s \in S$. Thus P^0 equates all and the only the elements of $IS^\star$ with the same interface. Fig.2(A) shows the sequence of partitions computed by the algorithm taking as initial state $IS = \{a, r\}$. It is important to note now that in the symbolic transition system of $IS^\star$ (Fig.2(B)) the only possibly redundant transition is $r \xrightarrow{\$^5,\alpha}_\eta s$ (because $r \xrightarrow{\varnothing,\alpha}_\eta t \vdash_{\mathcal{R}_\mathcal{N}} r \xrightarrow{\$^5,\alpha} t\$^5$). Thus, in order to check redundancy, at any iteration we have only to check if $t\$^5$ and s belong to the same block. In the initial partition all the states are equivalent since they all have the same interface (recall that all the marked nets presented in Section 3 have interface \$). In P^1 there are three blocks. The states a and r are in the same block because the transition $r \xrightarrow{\$^5,\alpha}_\eta s$ is redundant since s and $t\$^5$ belong to the same block in P^0. In the second iteration, the state $t\$^1$ is separated from $\{t\$^2, t\$^3, t\$^4, t\$^5, s\}$ because the former can perform $\xrightarrow{\varnothing,\beta}_\eta \{r, b\}$ while all the others cannot. Note that a and r are still in the same block because s and $t\$^5$ belong to the same block in P^1. In each of the following iteration, a state $t\$^i$ is separated from s. In P^6, the state $t\$^5$ is separated from s and thus in P^7 the states a and r are divided because the transition $r \xrightarrow{\$^5,\alpha}_\eta s$ is not redundant anymore. Then P^8 is equivalent to P^7 and the algorithm returns such partition.

$P^0 = \{a, b, r, s, t, t\$^1, t\$^2, t\$^3, t\$^4, t\$^5\}$
$P^1 = \{a, r\}\{b, t\}\{t\$^1, t\$^2, t\$^3, t\$^4, t\$^5, s\}$
$P^2 = \{a, r\}\{b, t\}\{t\$^1\}, \{t\$^2, t\$^3, t\$^4, t\$^5, s\}$
$P^3 = \{a, r\}\{b, t\}\{t\$^1\}, \{t\$^2\}\{t\$^3, t\$^4, t\$^5, s\}$
$P^4 = \{a, r\}\{b, t\}\{t\$^1\}, \{t\$^2\}\{t\$^3\}\{t\$^4, t\$^5, s\}$
$P^5 = \{a, r\}\{b, t\}\{t\$^1\}, \{t\$^2\}\{t\$^3\}\{t\$^4\}\{t\$^5, s\}$
$P^6 = \{a, r\}\{b, t\}\{t\$^1\}, \{t\$^2\}\{t\$^3\}\{t\$^4\}\{t\$^5\}\{s\}$
$P^7 = \{a\}\{r\}\{b, t\}\{t\$^1\}\{t\$^2\}\{t\$^3\}\{t\$^4\}\{t\$^5\}\{s\}$
$P^8 = \{a\}\{r\}\{b, t\}\{t\$^1\}\{t\$^2\}\{t\$^3\}\{t\$^4\}\{t\$^5\}\{s\}$

(A)



(B)

Fig. 2. (A)The partitions computed by `Symbolic-Partition-Refinement`$(\{a, r\})$.(B) The symbolic transition systems of $\{a, r\}^\star$.

In order to prove the soundness of our algorithm we define the *irredundant terminal sequence for the set of initial states* IS,

$$\sim^0_{IS} = \sim^0_\beta \restriction IS^\star \qquad \sim^{\kappa+1}_{IS} = \mathbf{IR}(\sim^\kappa_{IS}) \qquad \sim^\lambda_{IS} = \bigcap_{\kappa<\lambda} \sim^\kappa_{IS}$$

where $R \restriction A$ denotes the restriction of the relation R to the set A, $IS^\star$ is the closure of IS w.r.t. rules in Table 2.

The only difference with respect to the irredundant terminal sequence is in the first element. Here instead of taking the whole state space of $\mathcal{I}$, we restrict to $IS^\star$. The following theorem guarantees that this is enough in order to characterize the restriction of the irredundant terminal sequence to $IS^\star$. This is not trivial and it strongly relies on the fact that we close IS w.r.t. the rule (RD) in Table 2. Indeed whenever we remove such rule, it does not hold anymore.

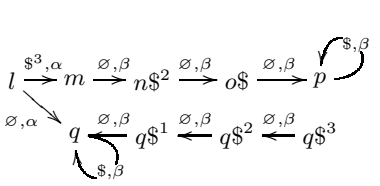
Theorem 4. *$\forall \kappa \in \mathcal{O}$, $\sim^\kappa_{ND} \restriction IS^\star = \sim^\kappa_{IS}$.*

Theorem 5. *If $\sim^\kappa_{IS} = \sim^{\kappa+1}_{IS}$, then $\forall k' \geq k+1$, $\sim^\kappa_{IS} = \sim^{\kappa'}_{IS}$.*

Corollary 1. *If $IS^\star$ is finite, then the algorithm terminates and the resulting partition equates all and only saturated bisimilar states.*

Since the algorithm applies to a lot of different formalisms, it is hard to provide a meaningful complexity analysis. However, we want to remark that the operation of checking redundancy is not expensive, since all the possible redundancies can be computed during the initialization (when using the rule (RD) of Table 2) and at any iteration, only those redundancies must be checked. Instead, the closure $IS^\star$ can be much larger than the set of reachable states (that is used by the ordinary partition refinement). Even worst, in our theory, nothing guarantees that if the set of reachable states (through the SCTS) is finite then also the closure $IS^\star$ is finite. However, we conjecture that this holds for many formalisms. The following proposition states that this holds in our running example.

Proposition 7. *Let $\mathcal{N}$, η and $\mathcal{R}_\mathcal{N}$ be the context interactive system, the symbolic transition system and the inference system for open nets that we have*



$$
\begin{aligned}
P^0 &= \{l, p, q, q\$^1, o\$, q\$^2, n, q\$^3, m\}\\
P^1 &= \{l\}\{p, q\}\{q\$^1, o\$, q\$^2, n, q\$^3, m\}\\
P^2 &= \{l\}\{p, q\}\{q\$^1, o\$\}\{q\$^2, n, q\$^3, m\}\\
P^3 &= \{l\}\{p, q\}\{q\$^1, o\$\}\{q\$^2, n\}\{q\$^3, m\}\\
P^4 &= \{l\}\{p, q\}\{q\$^1, o\$\}\{q\$^2, n\}\{q\$^3, m\}
\end{aligned}
$$

Fig. 3. The partions computed by `Symbolic-Partition-Refinement({`l`})`

introduced in Section 3. Let $\langle N, m\rangle$ be a marked open net. If the symbolic transition system of $\langle N, m\rangle$ is finite, then also the closure w.r.t. rules in Table 2 is finite.

6.3 Minimal Symbolic Automaton

Now we introduce minimal symbolic automata, i.e. automata having only irredundant symbolic transitions. We show that they are canonical representatives for equivalence classes of saturated bisimilar states. Moreover, we provide an algorithm to compute them. Hereafter, given an S sorted family of sets $X = \{X_s \mid s \in S\}$ and an S sorted family of equivalence relations $R = \{R_s \subseteq X_s \times X_s \mid s \in S\}$, we write $X_{s\,|R}$ to mean the set of equivalence classes of X_s w.r.t. R_s and for each $p \in X$, $[p]_R$ to mean the equivalence class of p w.r.t. R.

Definition 12 (Minimal Symbolic Automaton). *Let $\mathcal{I} = \langle (S, \Sigma), \mathbb{A}, O, tr\rangle$ be a context interactive system, β a symbolic transition system and $\mathcal{R}$ an inference system. Let p be a state of $\mathcal{I}$ and $\{p\}^\star = \{\{p\}^\star_s \mid s \in S\}$ be the S sorted family of sets of states obtained by closing $\{p\}$ with the rules in Table 2. The* minimal symbolic automaton of p *(denoted by $MSA(p)$) is a triple $\langle i, M, tr_M\rangle$:*

- *the initial state i is equal to $[p]_{\sim^S}$,*
- *$M = \{M_s \subseteq \{p\}^\star_{s\,|\sim^S} \mid s \in S\}$ is an S indexed family of set of equivalence classes of $\sim^S$,*
- *$tr_M \subseteq M \times \Sigma \times O \times M$ is a transition relation,*

defined according to the following two rules.

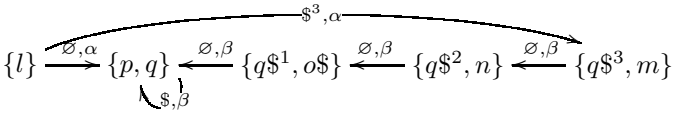
$$\frac{p \in A_s}{[p]_{\sim^S} \in M_s} \qquad \frac{[q]_{\sim^S} \in M \quad q \xrightarrow{c,o}_\beta r \text{ is irredundant in } \sim^S \quad r \in A_s}{[q]_{\sim^S} \xrightarrow{c,o}_M [r]_{\sim^S} \qquad [r]_{\sim_S} \in M_s}$$

The leftmost rule states that the equivalence class of the initial state p belongs to the states of the minimal automaton. The other rule adds all the equivalence classes that are reachable from p trough symbolic irredundant transitions. Notice that in the minimal automaton for standard bisimilarity (Def.1) the set of states consisted of all the equivalence classes of reachable states, and thus in order to compute the minimal automata, we just needed to quotient the set of reachable states. For minimal symbolic automata we have also to remove all those states that are not reachable through irredundant symbolic transitions. As an example

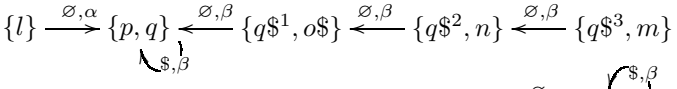
Algorithm 3. `Symbolic-Minimization(`p`)`

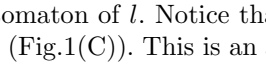
1. P :=`Symbolic-Partition-Refinement(`$\{p\}$`)`,
2. Quotient $\{p\}^\star$ w.r.t. P,
3. Remove the redundant transitions,
4. Remove the states that are not reachable.

consider the symbolic transition system of l (Fig.1(C)). Fig.3 shows the closure $\{l\}^\star$ and the partitions computed by `Symbolic-Partition-Refinement(`$\{l\}$`)`. The minimal automata of l can be constructed as follows. First, we quotient the states in $\{l\}^\star$ with respect to the partition P^4 returned by the algorithm.



Then we remove the redundant transitions.



Finally we take the set of states reachable from l: 

. This is the minimal symbolic automaton of l. Notice that it is isomorphic to the symbolic transition system of a (Fig.1(C)). This is an alternative proof of $a \sim^S l$. Indeed, for minimal symbolic automata, analogously to minimal automata, two states p and q are saturated bisimilar if and only if their minimal symbolic automata are isomorphic, where by isomorphism we mean a bijection on states that preserves sorts, transitions and initial states.

Proposition 8. *$p \sim^S q$ if and only if $MSA(p)$ is isomorphic to $MSA(q)$.*

7 Conclusions and Related Works

Relying on the framework of [4], we have introduced a symbolic partition refinement algorithm that allows to efficiently check saturated bisimilarity. Our approach is absolutely general and it can be applied to many formalisms. However, when considering nominal calculi where systems are able communicate names, the symbolic transition system is often infinite. Indeed, every time that a system generates a new name and extrudes it, the system goes in a new state that is different from all the previous. HD-Automata [16] are peculiar LTSs that allow to garbage collect names and avoid this other source of infiniteness. As future work, we will extend our framework to HD-Automata, so that we will be able to handle systems that generates infinitely many names. In particular we conjecture that this algorithm will generalize both [19] and [15] that provide a partition refinement algorithm for open [20] and asynchronous [1] bisimilarity.

Indeed, both our approach and [19,15] rely on *irredundant transitions*. In all these algorithms, first the closure of the set of initial states is computed by

adding, not only the reachable states, but also those states that are needed to check redundancy. Then, at any iteration, only irredundant transitions are considered. In [19], the closure is called *saturated state graph* and it is computed analogously to our approach. Instead, in [15], the closure is computed by adding *negative transitions* whenever there is a possible redundancy. Roughly, if $p \overset{a}{\rightsquigarrow} q$ is a negative transition, then a transition $p \xrightarrow{a} q'$ is redundant whenever the arriving state q and q' are the same. A novel notion of bisimilarity is introduced for these kind of transition systems, but it fails to be transitive. In our context interactive systems we just rely on the algebraic structure of contexts and irredundant bisimilarity coincides with the saturated one.

Moreover, the functions Φ and Φ_A, that are used during the iteration of the algorithms in [19,15], are not monotone and, as a consequence, the convergency of the corresponding terminal sequences have to be proven by hand. Instead in our approach the function **IR** generates exactly the same sequence of saturated bisimilarity and thus convergence and coincidence with saturated bisimilarity are for free. Moreover, we have shown that the correspondence between irredundant bisimilarity and saturated bisimilarity is not by chance, but because **IR** and **SAT** behaves exactly in the same way when restricted to congruences.

References

1. Amadio, R.M., Castellani, I., Sangiorgi, D.: On bisimulations for the asynchronous π-calculus. In: Sassone, V., Montanari, U. (eds.) CONCUR 1996. LNCS, vol. 1119, pp. 147–162. Springer, Heidelberg (1996)
2. Baldan, P., Corradini, A., Ehrig, H., Heckel, R.: Compositional semantics for open Petri nets based on deterministic processes. M.S.C.S 15(1), 1–35 (2005)
3. Baldan, P., Corradini, A., Ehrig, H., Heckel, R., König, B.: Bisimilarity and behaviour-preserving reconfiguration of open petri nets. In: Mossakowski, T., Montanari, U., Haveraaen, M. (eds.) CALCO 2007. LNCS, vol. 4624, pp. 126–142. Springer, Heidelberg (2007)
4. Bonchi, F., Montanari, U.: Symbolic semantics revisited. In: Amadio, R. (ed.) FOSSACS 2008. LNCS, vol. 4962, pp. 395–412. Springer, Heidelberg (2008)
5. Bonchi, F., Montanari, U.: Minimization algorithm for symbolic bisimilarity. Technical Report TR-08-27, Department of Informatics, University of Pisa (2008)
6. Cardelli, L., Gordon, A.D.: Mobile ambients. T.C.S. 240(1), 177–213 (2000)
7. Fernandez, J.C., Mounier, L.: "on the fly" verification of behavioural equivalences and preorders. In: Larsen, K.G., Skou, A. (eds.) CAV 1991. LNCS, vol. 575, pp. 181–191. Springer, Heidelberg (1992)
8. Ferrari, G.L., Gnesi, S., Montanari, U., Pistore, M., Ristori, G.: Verifying mobile processes in the hal environment. In: Vardi, M.Y. (ed.) CAV 1998. LNCS, vol. 1427, pp. 511–515. Springer, Heidelberg (1998)
9. Hennessy, M., Lin, H.: Symbolic bisimulations. T.C.S. 138(2), 353–389 (1995)
10. Honda, K., Tokoro, M.: An object calculus for asynchronous communication. In: America, P. (ed.) ECOOP 1991. LNCS, vol. 512, pp. 133–147. Springer, Heidelberg (1991)
11. Kanellakis, P.C., Smolka, S.A.: Ccs expressions, finite state processes, and three problems of equivalence. Information and Computation 86(1), 43–68 (1990)

12. Merro, M., Zappa Nardelli, F.: Bisimulation proof methods for mobile ambients. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, pp. 584–598. Springer, Heidelberg (2003)
13. Milner, R.: Communicating and Mobile Systems: the π-Calculus. Cambridge University Press, Cambridge (1999)
14. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, i and ii. Information and Computation 100(1), 1–40 (1992)
15. Montanari, U., Pistore, M.: Finite state verification for the asynchronous pi-calculus. In: Cleaveland, W.R. (ed.) TACAS 1999. LNCS, vol. 1579, pp. 255–269. Springer, Heidelberg (1999)
16. Montanari, U., Pistore, M.: An introduction to history dependent automata. Electr. Notes Theor. Comput. Sci. 10 (1997)
17. Montanari, U., Sassone, V.: Dynamic congruence vs. progressing bisimulation for ccs. Fundamenta Informaticae 16(1), 171–199 (1992)
18. Paige, R., Tarjan, R.E.: Three partition refinement algorithms. SIAM J. Comput. 16(6), 973–989 (1987)
19. Pistore, M., Sangiorgi, D.: A partition refinement algorithm for the π-calculus. Information and Computation 164(2), 264–321 (2001)
20. Sangiorgi, D.: A theory of bisimulation for the π-calculus. Acta Informatica 33(1), 69–97 (1996)
21. Victor, B., Moller, F.: The mobility workbench - a tool for the pi-calculus. In: Dill, D.L. (ed.) CAV 1994. LNCS, vol. 818, pp. 428–440. Springer, Heidelberg (1994)