Simulation as Coarsest Partition Problem

Raffaella Gentilini, Carla Piazza, and Alberto Policriti

Dip. di Matematica e Informatica,
Università di Udine
Via Le Scienze 206, 33100 Udine - Italy.
{gentilini|piazza|policriti}@dimi.uniud.it

Abstract. The problem of determining the coarsest partition stable with respect to a given binary relation, is known to be equivalent to the problem of finding the maximal *bisimulation* on a given structure. Such an equivalence has suggested efficient algorithms for the computation of the maximal bisimulation relation.
In this paper the *simulation* problem is rewritten in terms of coarsest stable partition problem allowing a more algebraic understanding of the simulation equivalence. On this ground, a new algorithm for deciding simulation is proposed. Such a procedure improves on either space or time complexity of previous simulation algorithms.

1 Introduction

In this work we deal with the problem of determining the so-called *simulation* relation on a given (Kripke) labeled structure $G = \langle N, E, \Sigma \rangle$. Such a problem consists, given G, in getting to the (unique) maximal binary relation $\leq_s$ such that

1. a node can simulate another one only if they have the same label;
2. if a node m simulates another node n, then any successor of n can be simulated by some successor of m.

On the ground of the above definition the notion of *sim-equivalence* $\equiv_s$ is introduced: $m \equiv_s n$ iff $m \leq_s n$ and $n \leq_s m$. In particular, we are interested in computing the similarity quotient $G/\equiv_s$.

The notion of simulation is very similar (less demanding, in fact) to the notion of *bisimulation*: an extremely pervasive idea proposed in many different fields, such as Modal Logic, Concurrency Theory, Set Theory, Automata Theory, etc. (cf. [1,15,11,14]). Two nodes m and n are bisimilar ($m \equiv_b n$) iff:

1. they have the same label;
2. any successor of n is bisimilar to some successor of m, and vice-versa.

If the naturalness of the concept of bisimulation explains its large usage—especially in connection with circular structures—, from a computational point of view the main reasons for its fortune and for its best solution lie in the possibility of re-formulating a bisimulation problem in purely (elementary) algebraic terms:

J.-P. Katoen and P. Stevens (Eds.): TACAS 2002, LNCS 2280, pp. 415–430, 2002.

a bisimulation is the coarsest partition finer than an input one and *stable* with respect to the relation E of the graph (a partition is stable with respect to E iff any of its classes is either sent by E entirely within or entirely outside any other of its classes).

Both bisimulation and simulation (as well as other possible relations of the same sort) are used in order to *simplify* the input structure by collapsing all the nodes of the same equivalence class into a single node. As explained in [13] "in many cases, neither trace equivalence nor bisimilarity, but similarity is the appropriate abstraction for computer-aided verification ...". In the case of finite-state systems the similarity quotient $G/\equiv_s$ can be computed in polynomial time, while this is not the case for trace equivalence quotient. In the case of infinite-state systems, finitely represented using hybrid automaton and other formalisms, the similarity quotients can be computed symbolically and in many cases the quotients are finite (see [13]). Since the conditions in the definition of simulation are weaker than the ones in the definition of bisimulation, simulation provides a better space reduction than bisimulation (i.e., $|G/\equiv_s| \leq |G/\equiv_b|$) and it is still adequate for the verification of all the formulae of the branching temporal logic without quantifiers switches (e.g. the formulae of ACTL*, see [16]).

Several polynomial-time algorithms for computing similarity quotients on finite graphs have been proposed: the ones presented in [2], [5], and [6] achieve time complexities of the orders $O(|N|^6|E|)$, $O(|N|^4|E|)$, and $O(|E|^2)$, respectively. A simulation procedure running in $O(|N||E|)$ time was independently discovered in [13] and [3]. All of the algorithms just mentioned ([2], [5], [6], [3], [13]) obtain the similarity quotient as a by-product of the computation of the entire similarity relation on the set of states N. Their space complexity is then limited from below by $O(|N|^2)$.

Recently Bustan and Grumberg in [4] and Cleaveland and Tan in [7] improved the above results.

The procedure by Bustan and Grumberg [4] gives in output the quotient structure with respect to $\equiv_s$ and the simulation relation among the classes of $M = N/_{\equiv_s}$ without computing the entire simulation on N. Hence, its space requirements (often more critical, especially in the field of verification) depends on the size of M and are lower than the ones of the algorithms in [2], [5], [6], [3], and [13]. In more detail, the so-called *Partitioning Algorithm* described in [4] uses only $O(|M|^2 + |N|\log(|M|))$ space whereas its time complexity is rather heavy: it is $O(|M|^4(|E| + |M|^2) + |M|^2|N|(|N| + |M|^2))$.

The procedure in [7] combines the fix-point calculation techniques in [3] and [13] with the bisimulation-minimization algorithm in [17]. A system, G_2, is determined being or not capable of simulating the behavior of G_1, by interleaving the minimization via bisimulation of the two systems with the computation of the set of classes in G_2 able to simulate each class in G_1. The time complexity achieved is $O(|B_1||B_2| + |E_1|\log(|N_1|) + |B_1||E_2| + |\varepsilon_1||B_2|)$, where ε_i and B_i represent the bisimulation reduced relation and states'space of T_i. Compared with the time complexities of [13] and [3], the latter expression have many occurrences of $|N_i|$ and $|E_i|$ replaced with $|B_i|$ and $|\varepsilon_i|$. Indeed, the experimental

results in [7] prove that the procedure by Tan and Cleaveland outperform the ones in [13] and [3]. The space complexity of [7] depends on the product of the sizes of the two bisimulation quotients involved. Being bisimulation finer than simulation, such a space requirement may be more demanding than the one in [4].

In our work we start by observing that a simulation equivalence can be seen as a bisimulation in which the condition on the children is weaker than the condition on the parents: in order to have $m \equiv_s n$ is sufficient that a back-and-forth condition involving $\leq_s$ is satisfied. In the case of $\equiv_b$, instead, in order to have $m \equiv_b n$ any child of m (resp. n) must be in the same relation $\equiv_b$ with some of n's (resp. m's) children. This fact is, ultimately, the reason why on the one hand, the computational steps for determining a simulation must compute (successive approximations of) both $\equiv_s$ and $\leq_s$ and, on the other hand, it is not easy to rephrase the notion of simulation in purely algebraic terms as for bisimulation.

Here we show that such a rephrasing is in fact possible and that the simulation problem can be rewritten in terms of a coarsest stable partition problem in which partitions are also equipped with an acyclic relation over their classes. The simulation quotient equipped with the partial order induced on it by $\leq_s$ is shown to correspond to the "coarsest partition-pair" stable w.r.t. a suitable condition. Such a characterization of $\equiv_s$ (as well as of $\leq_s$) gives new insight on to the algebraic properties of the simulation equivalence. Moreover, it underpins the designing of a new space-efficient, and not too costly in time, algorithm for determining M and the simulation among its classes.

The key idea in our approach is that of using, in a sequence of approximation steps determining M, a graph (the $\exists\forall$-structure, similar to the structures used in [8]) whose nodes are equivalence classes and whose edges are capable to convey all the necessary information present in the original structure G. Such information is captured in a very natural way: since in the approximations nodes are in fact classes of nodes, we introduce two kind of edges between classes α and α', corresponding to the case in which either *there is* an edge between an element in α and one in α', or to the case in which *all* nodes in α are sent in α' by E. The stability condition (to be proved) equivalent to the notion of simulation is the following: a class β simulates a class α only if whenever there is an $\exists$-edge between α and γ, then there exists a $\forall$-edge between β and δ simulating γ (cf. Figure 1).

The use of the above structure and the formulation of the simulation problem as a coarsest stable partition problem allows us to define an algorithm computing the simulation relation as follows:

1. [13] can be used on a suitable $\exists\forall$-structure to move from an approximation to the next one, without wasting space (moreover, at any step the partition Σ_i is modified);
2. the algorithm stops when stability is reached.

The space complexity of our algorithm is $O(|M|^2 + |N| \log(|M|))$, exactly the same as [4] with which our algorithm shares the general structure. However, due

to the use of [13] as subroutine, our time complexity turns out to be $O(|M|^2|E|)$, much better than $O(|M|^4(|E|+|M|^2)+|M|^2|N|(|N|+|M|^2))$ which is the time complexity of the algorithm in [4].

All the proofs of the claims in this paper can be found in [12] (available on the web).

2 Preliminaries

As far as the notions of *quasi order* and *partial order* are concerned, we refer to [9]. We will use Q^+ to refer to the transitive closure of a relation Q and Q^* to refer to its reflexive and transitive closure.

Definition 1. *A triple $G = \langle N, E, \Sigma\rangle$ is said to be a* labeled graph *if and only if $G^- = \langle N, E\rangle$ is a finite graph and Σ is a partition over N. We say that two nodes $n_1, n_2 \in N$ have the same* label *if they belong to the same class of Σ.*

Given a node $n \in N$ we will use $[n]_\Sigma$ (or $[n]$, if Σ is clear from the context) to denote the class of Σ to which n belongs.

Definition 2. *Let $G = \langle N, E, \Sigma\rangle$ be a labeled graph. A relation $\leq \subseteq N \times N$ is said to be a* simulation *over G if and only if:*

1. $n \leq m \rightarrow [n]_\Sigma = [m]_\Sigma$;
2. $(n \leq m \wedge nEn_1) \rightarrow \exists m_1(mEm_1 \wedge n_1 \leq m_1)$.

In this case we say that m simulates *n. We also say that m and n are* sim-equivalent *($m \equiv_s n$) if there exist two simulations $\leq_1$ and $\leq_2$, such that $n \leq_1 m$ and $m \leq_2 n$.*

A simulation over $G = \langle N, E, \Sigma\rangle$ is said to be *maximal* if it is maximal w.r.t. $\subseteq$.

Proposition 1. *Given a labeled graph $G = \langle N, E, \Sigma\rangle$ there always exists a unique maximal simulation $\leq_s$ over G and $\leq_s$ is a quasi order over N. The relation $\equiv_s$ (sim-equivalence) is an equivalence relation over N.*

Definition 3. *Given a labeled graph $G = \langle N, E, \Sigma\rangle$ the problem of* computing the similarity quotient *of G consists in computing the quotient $N/\equiv_s$, where $\equiv_s$ is the sim-equivalence over G.*

In [4] it has also been proved that there always exists a unique smallest labeled graph that is simulation equivalent to G, i.e. there is a unique way to put a minimum number of edges between the elements of $N/\equiv_s$ in order to obtain a labeled graph similar to G.

3 Simulation as Coarsest Partition Problem

In this section we introduce the *Generalized (Stable) Coarsest Partition Problem* (GCPP) which is the central notion in our approach. We call it *generalized* because we are not only going to deal with partitions to be refined (as in the classical coarsest partition problems [18,17,14]), but with *pairs* constituted of a partition and a relation over the partition. The equivalence of the similarity quotient problem and the GCPP will be proved at the end of this section.

Definition 4. *Let $G = \langle N, E\rangle$ be a finite graph. A* partition-pair *over G is a pair $\langle \Sigma, P\rangle$ in which Σ is a partition over N, and $P \subseteq \Sigma^2$ is a reflexive and acyclic relation over Σ.*

Given a labeled graph $G = \langle N, E, \Sigma\rangle$ an example of a partition-pair over G is given by the pair $\langle \Sigma, I\rangle$, where I is the identity relation over Σ.

Given two partitions Π and Σ, such that Π is finer than Σ, and a relation P over Σ, we use the notation $P(\Pi)$ to refer to the relation *induced* on Π by P, i.e.:

$$\forall \alpha\beta \in \Pi((\alpha, \beta) \in P(\Pi) \leftrightarrow \exists \alpha'\beta'((\alpha \subseteq \alpha' \wedge \beta \subseteq \beta' \wedge (\alpha', \beta') \in P)).$$

Denoting by $\mathcal{P}(G)$ the set of partition-pairs over G, we now introduce the partial order we need in order to be able to define the notion "$\langle \Sigma, P\rangle$ is *coarser* than $\langle \Pi, Q\rangle$".

Definition 5. *Let $\langle \Sigma, P\rangle, \langle \Pi, Q\rangle \in \mathcal{P}(G)$:*

$$\langle \Pi, Q\rangle \sqsubseteq \langle \Sigma, P\rangle \text{ iff } \Pi \text{ is finer than } \Sigma \text{ and } Q \subseteq P(\Pi).$$

The following definition introduces structures that in the algorithm, will suggest us the use of a subroutine based on the simulation algorithm by Henzinger, Henzinger, and Kopke in [13]. The use of such a subroutine guarantees a weak form of stability (see Definition 10) in the intermediate stages and is one of the keys in the improved time performance of our algorithm w.r.t. the one by Bustan and Grumberg [4].

Definition 6. *Let $G = \langle N, E\rangle$ be a graph and Π be a partition of N. The* $\exists$-quotient structure *over Π is the graph $\Pi_\exists = \langle \Pi, E_\exists\rangle$, where*

$$\alpha E_\exists \beta \quad \text{iff} \quad \exists n \exists m(n \in \alpha \wedge m \in \beta \wedge nEm).$$

The $\forall$-quotient structure *over Π is the graph $\Pi_\forall = \langle \Pi, E_\forall\rangle$, where*

$$\alpha E_\forall \beta \quad \text{iff} \quad \forall n(n \in \alpha \rightarrow \exists m(m \in \beta \wedge nEm)).$$

The $\exists\forall$-quotient structure *over Π is the structure $\Pi_{\exists\forall} = \langle \Pi, E_\exists, E_\forall\rangle$.*

Notice that $\alpha E_\forall \beta$ iff $\alpha \subseteq E^{-1}(\beta)$ and $\alpha E_\exists \beta$ iff $\alpha \cap E^{-1}(\beta) \neq \emptyset$. Similar notations (called *relation transformers* and *abstract transition relations*) were introduced in [8] in order to combine Model Checking and Abstract Interpretation.

We introduce here a definition, strongly connected to the definition of the quotient structures, that we will use later in the description of our algorithm (see Section 5).

Definition 7. *Given a graph, $G = \langle N, E\rangle$, and two partitions of N, Σ and Π, with Π finer than Σ, the* $\exists\forall$-induced quotient structure *over Π is the structure $\Sigma_{\exists\forall}(\Pi) = \langle \Pi, E^\Sigma_\exists(\Pi), E^\Sigma_\forall(\Pi)\rangle)$, where:*

$$\begin{array}{l} \alpha E^\Sigma_\exists(\Pi)\beta \text{ iff} \quad \alpha \cap E^{-1}(\beta) \neq \emptyset \\ \alpha E^\Sigma_\forall(\Pi)\beta \text{ iff} \quad \alpha E^\Sigma_\exists(\Pi)\beta \quad \wedge \quad \alpha \subseteq E^{-1}(\beta') \wedge \quad \beta \subseteq \beta' \in \Sigma \end{array}$$

with $\alpha, \beta \in \Pi$.

We are now ready to introduce the fundamental notion in the generalized coarsest partition problems, that is the notion of stability of a partition-pair w.r.t. a relation:

Definition 8. *Given a graph $G = \langle N, E\rangle$, we say that a partition-pair $\langle \Sigma, P\rangle$ over G is* stable *w.r.t. the relation E if and only if*

$$\forall \alpha, \beta, \gamma \in \Sigma((\alpha, \beta) \in P \wedge \alpha E_{\exists} \gamma \rightarrow \exists \delta \in \Sigma((\gamma, \delta) \in P \wedge \beta E_{\forall} \delta)). \tag{1}$$

Condition (1) in the previous definition is equivalent to:

$$\forall \alpha, \beta, \gamma \in \Sigma((\alpha, \beta) \in P \wedge \alpha \cap E^{-1}(\gamma) \neq \emptyset \rightarrow \exists \delta \in \Sigma((\gamma, \delta) \in P \wedge \beta \subseteq E^{-1}(\delta))). \tag{2}$$

The stability condition is exactly the condition which holds between the classes of $N/\equiv_s$: if $\alpha, \beta \in N/\equiv_s$ with $\alpha \leq_s \beta$ (i.e. all the elements of α are simulated by all the elements of β), and an element a in α reaches an element c in γ, then all the elements b of β must reach at least one element d which simulates c. Taking, in particular, all the maximal (w.r.t. $\leq_s$) elements d simulating c reached by elements in β, we have that all the elements in β reach a class δ which simulates c and, hence, which simulates γ.

In Figure 1 we give a graphical representation of the notion of stability using both the characterizations (1) and (2).

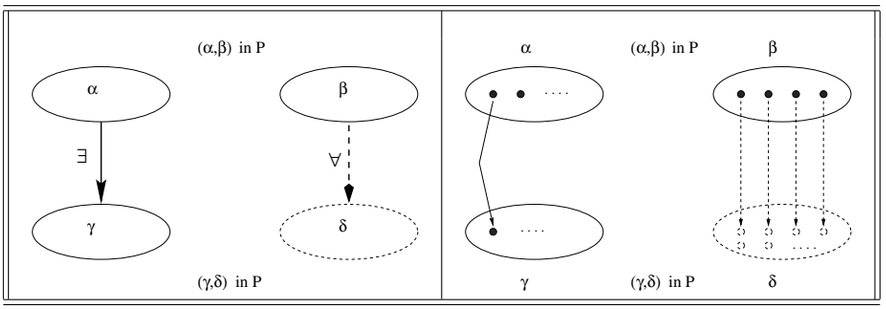

Fig. 1. The Stability Condition.

We will use the notion of stability of a partition-pair w.r.t. a relation in the definition of generalized coarsest partition problem, in the same way in which the notion of stability of a partition w.r.t. a relation is used in the definition of coarsest partition problem.

Definition 9 (Generalized Coarsest Partition Prob. – GCPP). *Let $G = \langle N, E\rangle$ be a graph and $\langle \Sigma, P\rangle$ be a partition-pair over G the* generalized coarsest partition problem *consists in finding a partition-pair $\langle M, \preceq\rangle$ such that:*
(a) $\langle M, \preceq\rangle \sqsubseteq \langle \Sigma, P^{+}\rangle$ *and* $\langle M, \preceq\rangle$ *is stable w.r.t. G;*
(b) $\langle M, \preceq\rangle$ *is $\sqsubseteq$-maximal satisfying (a).*

If $\langle \Pi, Q\rangle$ is a partition-pair which satisfies (a) only, we say that it is a stable refinement *of $\langle \Sigma, P\rangle$.*

Notice that in the above definition $\langle M, \preceq\rangle$ is a refinement of $\langle \Sigma, P^+\rangle$, while it can be the case that it is not a refinement of $\langle \Sigma, P\rangle$. In particular this happens always in case $\langle \Sigma, P\rangle$ is not stable while $\langle \Sigma, P^+\rangle$ is stable. In this case $\langle \Sigma, P^+\rangle$ itself is a solution of the GCPP. In general, the solution of the GCPP can always be found by a suitable sequence of splits (of classes) and adequate completion of the relation in order to take the newly generated classes into account.

Remark 1. Notice that it is important that $\preceq$ is reflexive (it is reflexive, since $\langle M, \preceq\rangle$ is a partition-pair). This is necessary in order to prove that the maximal solution is unique. Given $G = \langle N, E\rangle$ with $N = \{n_1, n_2\}$ and $E = \{(n_1, n_2)\}$, and $\langle \Sigma, P\rangle$ with $\Sigma = \{N\}$ and $P = \{(N, N)\}$. It holds that both $\langle \Sigma, \emptyset\rangle$ and $\langle \Pi, Q\rangle$ with $\Pi = \{\{n_1\}, \{n_2\}\}$ and $Q = \{(\{n_1\}, \{n_1\}), (\{n_2\}, \{n_2\}), (\{n_2\}, \{n_1\})\}$ are maximal solutions of the partition problem, but the first is not a partition-pair. Similarly, the acyclicity condition is important because otherwise we could coarsen the partition by merging all the classes which form cycles.

We prove that a GCPP has always a unique solution and then we spell out the connection between similarity quotient problems and GCPP's.

Theorem 1. *The GCPP over G and $\langle \Sigma, P\rangle$ has a unique solution $\langle M, \preceq\rangle$ and the relation $\preceq$ is a partial order over M.*

The proof of this theorem is carried out by proving that each generalized coarsest partition problem has a unique solution which can be determined by solving a similarity quotient problem.

Also the opposite direction is true: a similarity quotient problem can be solved using a generalized coarsest partition problem.

Theorem 2. *Let $G = \langle N, E, \Sigma\rangle$ be a labeled graph. Let $\langle M, \preceq\rangle$ be the solution of the GCPP over $G^- = \langle N, E\rangle$ and the partition-pair $\langle \Sigma, I\rangle$, where I is the identity relation over Σ. M is the simulation quotient of G and*

$$\forall n, m \in N \; n \leq_s m \text{ iff } [n]_M \preceq [m]_M.$$

Hence, in order to solve the problem of determining the similarity quotient of a labeled graph $G = \langle N, E, \Sigma\rangle$ we can, equivalently, solve the generalized coarsest partition problem over $\langle N, E\rangle$ and $\langle \Sigma, I\rangle$. If $\langle M, \preceq\rangle$ is the solution of the GCPP (i.e. the maximal refinement of $\langle \Sigma, I\rangle$ stable w.r.t. E), then the relation $\leq_{\langle M, \preceq\rangle}$ defined as

$$\forall n, m \in N (n \leq_{\langle M, \preceq\rangle} m \text{ iff } [n]_M \preceq [m]_M)$$

is the maximal simulation over G, and M is the partition which corresponds to the sim-equivalence $\equiv_s$ (c.f. Definition 2).

4 Computing a Solution to the GCPP

We now introduce an operator σ mapping partition-pairs into partition-pairs, which will turn out to be the engine of our algorithm. The results in this section

will allow us to conclude that a procedure which computes σ can be used to solve GCPP's and, hence, to compute similarity quotients: it is only necessary to iterate the computation of σ at most $|M|$ times.

In particular, the operator σ is defined in such a way that it refines the partition-pair $\langle \Sigma, P\rangle$ obtaining a partition-pair $\langle \Pi, Q\rangle$ which is *more stable* than $\langle \Sigma, P\rangle$ and is never finer than the solution of the GCPP over $\langle \Sigma, P\rangle$. In the first condition of σ we impose to split the classes of Σ which do not respect the stability condition w.r.t. themselves: if a class α is such that $\alpha E_\exists \gamma$ and it does not exists a class δ such that $(\gamma, \delta) \in P$ and $\alpha E_\forall \delta$, then the pair (α, α) does not respect the stability condition, hence we must split α. The first condition is used to build Π, and then the second and the third conditions are in σ used to define Q using the Π already obtained. Intuitively, the second and the third conditions allow to obtain Q from P by starting from $P(\Pi)$ and removing the minimum number of pairs which contradict stability.

Definition 10. *Let $G = \langle N, E\rangle$ be a graph and $\langle \Sigma, P\rangle$ be a partition-pair over G, the partition-pair $\langle \Pi, Q\rangle = \sigma(\langle \Sigma, P\rangle)$ is defined as:*
(1σ) Π is the coarsest partition finer than Σ such that

$$(a)\ \forall \alpha \in \Pi\ \forall \gamma \in \Sigma(\alpha E_\exists \gamma \rightarrow \exists \delta \in \Sigma((\gamma, \delta) \in P \wedge \alpha E_\forall \delta));$$

(2σ) Q is the maximal relation over Π such that $Q \subseteq P(\Pi)$ and if $(\alpha, \beta) \in Q$, then:

$$(b)\ \forall \gamma \in \Sigma(\alpha E_\forall \gamma \rightarrow \exists \gamma' \in \Sigma((\gamma, \gamma') \in P \wedge \beta E_\exists \gamma'))\quad \textit{and}$$
$$(c)\ \forall \gamma \in \Pi(\alpha E_\forall \gamma \rightarrow \exists \gamma' \in \Pi((\gamma, \gamma') \in Q \wedge \beta E_\exists \gamma')).$$

By abuse of notation we use $E_\exists$ and $E_\forall$ also when the classes belong to different partitions.

Condition (a) in (1σ) imposes to opportunely split the classes of the partition Σ: these splits are forced by the fact that we are looking for a partition-pair *stable* w.r.t. the relation of the given graph and we know that in each partition-pair $\langle \Sigma, P\rangle$ the second component is reflexive (i.e. $\forall \alpha \in \Sigma(\alpha, \alpha) \in P$). Using condition (b) in (2σ), together with condition (a) in (1σ) and exploiting the fact that P is acyclic, it is possible to prove that Q is acyclic: the acyclicity of P ensures that a cycle could arise only among classes of Π which are all contained in a unique class of Σ, then using condition (a) in (1σ) and (b) in (2σ) we obtain a contradiction. Condition (c) is fundamental, together with condition (a), in order to obtain the result in Theorem 4: if it holds that $\alpha E_\forall \gamma$, then *no matter how we split* α one of the subclasses generated from α has a chance (in the solution of GCPP) to be in relation with at least one of the subclasses generated from β only if $\beta E_\exists \gamma'$ and γ is in relation with γ'. The following results guarantee the correctness of σ and, hence, of our approach.

Theorem 3. *Let $G = \langle N, E\rangle$ be a graph and $\langle \Sigma, P\rangle$ be a partition-pair over G. There always exists a unique partition-pair $\langle \Pi, Q\rangle$ which satisfies the conditions in Definition 10, i.e. σ is always uniquely defined.*

Now that we have obtained that given a partition-pair $\langle \Sigma, P\rangle$, there exists a unique $\sigma(\langle \Sigma, P\rangle)$, we can link fix-points of the operator σ with solutions of GCPP's.

Lemma 1. *Let $\langle \Sigma, P\rangle$ and $\langle \Pi, Q\rangle$ be two partition-pairs and let $\langle M, \preceq\rangle$ be the solution of the GCPP over the graph G and $\langle \Sigma, P\rangle$. If $\langle M, \preceq\rangle \sqsubseteq \langle \Pi, Q\rangle$, then $\langle M, \preceq\rangle \sqsubseteq \sigma(\langle \Pi, Q\rangle)$*

Theorem 4. *Let $G = \langle N, E\rangle$ be a graph and $\langle \Sigma, P\rangle$ a partition-pair over G with P transitive. Let $\langle M, \preceq\rangle$ be the solution of the GCPP over G and $\langle \Sigma, P\rangle$. If i is such that $\sigma^i(\langle \Sigma, P\rangle) = \langle \Sigma_i, \mathsf{P}_i\rangle$ and $\sigma^{i+1}(\langle \Sigma, P\rangle) = \langle \Sigma_i, \mathsf{P}_{i+1}\rangle$, then $\mathsf{P}_{i+1} = \mathsf{P}_i$ and $\langle \Sigma_i, \mathsf{P}_i\rangle = \langle M, \preceq\rangle$.*

The meaning of this theorem is that if $\langle \Sigma, P\rangle$ is a partition-pair over a graph G such that P is transitive, then there exists an i such that $\sigma^i(\langle \Sigma, P\rangle) = \sigma^{i+1}(\langle \Sigma, P\rangle)$, and $\sigma^i(\langle \Sigma, P\rangle)$ is the solution of the GCPP over G and $\langle \Sigma, P\rangle$. In particular, when we iteratively apply the operator σ until we reach a fix-point, at each iteration we refine the partition and we remove pairs from the relation. What we proved in the above theorem is that it is never the case that there exists an iteration in which we do not refine the partition, but only remove pairs from the relation. In a certain sense this means that the two conditions (b) and (c) we gave in (2σ) to remove pairs are *optimal*. This property gives us the upper bound to the index i which is at most $|\Sigma_i|$, which in the worst case is $O(|N|)$.

Corollary 1. *The solution $\langle M, \preceq\rangle$ of the GCPP over a graph G and a partition-pair $\langle \Sigma, P\rangle$ can be computed using at most $|M|$ times the operator σ.*

Notice that, given a graph G and a partition-pair $\langle \Sigma, P\rangle$, the solution of the GPPC over G and $\langle \Sigma, P\rangle$ corresponds to the solution of the GPPC over G and $\langle \Sigma, P^+\rangle$.

The characterizations obtained in this section allow us to conclude that if we are able to define a procedure which, given a partition-pair $\langle \Sigma, P\rangle$ computes $\sigma(\langle \Sigma, P\rangle)$, then we can use it to solve GCPP's and hence to compute similarity quotients. In particular we recall that given a similarity quotient problem over a labeled graph $G = \langle N, E, \Sigma\rangle$, in order to solve it it is sufficient solve the GCPP over G and $\langle \Sigma, I\rangle$ (I is trivially transitive).

Moreover, the last corollary ensures that it will be necessary to iterate the procedure which computes σ at most $|M|$ times. This is a first improvement on the time complexity w.r.t. the algorithm presented in [4]: in a certain sense [4] computes an operator which refines the partition-pair less than σ, and hence it is possible that the computation has to be iterated up to $|M|^2$ times.

In the next section we present the procedure which computes σ.

5 The Algorithm

In this section we outline an algorithm which solves the generalized coarsest partition problem we have presented in the previous section. The **Stable Simulation Algorithm** takes as input a graph $G = \langle N, E\rangle$ and a partition-pair $\langle \Sigma, P\rangle$, with P transitive, calls the two functions **Refine** and **Update** until a fix-point is reached, and returns the partition-pair $\langle M, \preceq\rangle$ which is the solution of the GCPP over G and $\langle \Sigma, P\rangle$. In order to solve the GCPP over G and $\langle \Sigma, P\rangle$

with P not transitive, it is sufficient to first compute P^+ and the cost of this operation is $O(\Sigma^3)$ which, hence, does not affect the global cost of our algorithm. The function **Refine** takes as input a partition-pair $\langle \Sigma_i, P_i \rangle$ and returns the partition Σ_{i+1} which is the coarsest that satisfies the condition (a) in (1σ) of Definition 10. The function **Update** takes as input a partition-pair $\langle \Sigma_i, P_i \rangle$ and the refinement Σ_{i+1} and it produces the reflexive and acyclic relation over Σ_{i+1} which is the largest that satisfies conditions (b) and (c) in (2σ) of Definition 10. In particular, at the end of each iteration of the while-loop in the **Stable Simulation Algorithm** we have that $\langle \Sigma_{i+1}, P_{i+1} \rangle = \sigma(\langle \Sigma_i, P_i \rangle)$. It is immediate to

```
Stable Simulation Algorithm(⟨N, E⟩, ⟨Σ, P⟩)
    change := ⊤;
    i := 0;
    while change do
                        change := ⊥;
                        Σ_{i+1} := Refine(Σ_i, P_i, change);
                        P_{i+1} := Update(Σ_i, P_i, Σ_{i+1});
                        i := i + 1;
```

Fig. 2. The **Stable Simulation Algorithm**.

see that the **Refine** function works exactly as described in the proof of Theorem 3 (see [12]), and hence it produces the partition which is the first element of $\sigma(\Sigma_i, \mathsf{P}_i)$.

Corollary 2. *If* $\sigma(\langle \Sigma_i, \mathsf{P}_i \rangle) = \langle \Pi, Q \rangle$*, then* $\Pi = \Sigma_{i+1}$*.*

The deletion of "wrong" pairs is performed by **Update** through two calls to the function **New_HHK**, which is a version of [13] adapted to our purposes here. The function **Update** removes pairs from $\mathsf{P}_i(\Sigma_{i+1}) = \mathsf{Ind}_{i+1}$ in order to obtain the relation P_{i+1} which satisfies condition (2σ) of Definition 10. In particular Ref_{i+1} (obtained after the first call to **New_HHK** only) satisfies the first of the two conditions but not necessarily the second one.

Notice that the space complexity of the calls to **New_HHK** remains limited since they are made on quotient stuctures. This function is based on the use of the two structures $\Sigma_{i\exists\forall}(\Sigma_{i+1})$ (cf. Definition 7) and $\Sigma_{i+1\exists\forall}$ (cf. Definition 6), and on the following equivalent formulation of condition (2σ).

Proposition 2. *Let* $G = \langle N, E \rangle$ *be a graph,* $\langle \Sigma, P \rangle$ *be a partition-pair and* Π *be a partition finer than* Σ*.* Q *satisfies* (2σ) *of Definition 10 if and only if* Q *is the maximal relation over* Π *such that* $Q \subseteq P(\Pi)$ *and if* $(\alpha, \beta) \in Q$*, then:*

$$\forall \gamma \in \Pi(\alpha E^{\Sigma}_{\forall}(\Pi)\gamma \rightarrow \exists \gamma' \in \Pi((\gamma, \gamma') \in P(\Pi) \wedge \beta E^{\Sigma}_{\exists}(\Pi)\gamma')) \wedge$$
$$\forall \gamma \in \Pi(\alpha E_{\forall}\gamma \rightarrow \exists \gamma' \in \Pi((\gamma, \gamma') \in Q \wedge \beta E_{\exists}\gamma')),$$

where $E^{\Sigma}_{\forall}(\Pi)$ *and* $E^{\Sigma}_{\forall}(\Pi)$ *are the edges of the* $\exists\forall$*-induced quotient structure, while* $E_{\exists}$ *and* $E_{\forall}$ *are the edges of the* $\exists\forall$ *quotient structure.*

Refine$(\Sigma_i, \mathsf{P}_i, \mathsf{change})$

$\Sigma_{i+1} := \Sigma_i$;
for each $\beta \in \Sigma_{i+1}$ do $\mathsf{Stable}(\beta) := \emptyset$;
for each $\alpha \in \Sigma_i$ do $\mathsf{Row}(\alpha) := \{\gamma \mid (\alpha, \gamma) \in \mathsf{P}_i)\}$;
Let Sort be a topological sorting of $\langle \Sigma_i, \mathsf{P}_i \rangle$;
while $\mathsf{Sort} \neq \emptyset$ do
    $\alpha := dequeue(\mathsf{Sort})$;
    $A := \emptyset$;
    $\mathsf{Split}(\alpha) := \{\beta \in \Sigma_{i+1} \mid \beta E_\exists \alpha\}$;
    for each $\beta \in \mathsf{Split}(\alpha), \mathsf{Stable}(\beta) \cap \mathsf{Row}(\alpha) = \emptyset$ do
        $\beta_1 := \beta \cap E^{-1}(\alpha)$;
        $\beta_2 := \beta \setminus \beta_1$;
        if $\beta_2 \neq \emptyset$ then $\mathsf{change} := \top$;
        $\Sigma_{i+1} := \Sigma_{i+1} \setminus \{\beta\}$;
        $A := A \cup \{\beta_1, \beta_2\}$;
        $\mathsf{Stable}(\beta_1) := \mathsf{Stable}(\beta) \cup \{\alpha\}$;
        $\mathsf{Stable}(\beta_2) := \mathsf{Stable}(\beta)$;
    $\Sigma_{i+1} := \Sigma_{i+1} \cup A$;
    $\mathsf{Sort} := \mathsf{Sort} \setminus \{\alpha\}$;
return Σ_{i+1}

Fig. 3. The **Refine** Function.

Update$(\Sigma_i, \mathsf{P}_i, \Sigma_{i+1})$

$\mathsf{Ind}_{i+1} := \{(\alpha_1, \beta_1) \mid \alpha_1, \beta_1 \in \Sigma_{i+1}, \alpha_1 \subseteq \alpha, \beta_1 \subseteq \beta (\alpha, \beta) \in \mathsf{P}_i\}$;
$\Sigma_{i\exists\forall}(\Sigma_{i+1}) := \langle \Sigma_{i+1}, E_\exists^{\Sigma_i}(\Sigma_{i+1}), E_\forall^{\Sigma_i}(\Sigma_{i+1}) \rangle$;
$\mathsf{Ref}_{i+1} := \mathbf{New\_HHK}(\Sigma_{i\exists\forall}(\Sigma_{i+1}), \mathsf{Ind}_{i+1}, \bot)$;
$\Sigma_{(i+1)\exists\forall} := \langle \Sigma_{i+1}, E_\exists, E_\forall \rangle$;
$\mathsf{P}_{i+1} := \mathbf{New\_HHK}(\Sigma_{(i+1)\exists\forall}, \mathsf{Ref}_{i+1}, \top)$;
return P_{i+1}

Fig. 4. The **Update** Function.

In Figure 5 we present the conditions described in Proposition 2 on the $\exists\forall$-induced quotient structure and on the $\exists\forall$-quotient structure. As a consequence of Theorem 4 these conditions are weaker than the stability condition.

The computation performed by Update correspond to determine the largest relation included in P_i and satisfying conditions (2σ), thereby getting us closer to stability. Such a computation is proved correct as a fix-point computation of an operator τ defined as follows:

Definition 11. *Let $D = \langle S, R_1, R_2 \rangle$ be such that $R_2 \subseteq R_1$, and let $K \subseteq S \times S$. We define $\tau_D(K) = K \setminus \{(b, c) \mid \exists a(bR_2a \wedge \forall d(cR_1d \rightarrow (a, d) \notin K))\}$. We use $Fix(\tau_D)(K)$ to denote the greatest fix-point of τ_D smaller than K.*

Corollary 3. *Let $G = \langle N, E \rangle$ be a graph, and $\langle \Sigma, P \rangle$ be a partition-pair. If $\sigma(\langle \Sigma, P \rangle) = \langle \Pi, Q \rangle$, then $Q = Fix(\tau_{\Pi_{\exists\forall}})(\tau_{\Sigma_{\exists\forall}(\Pi)}(P(\Pi)))$.*

Now we complete the connection between the operator τ and the function **New_HHK** presented in Figure 6.

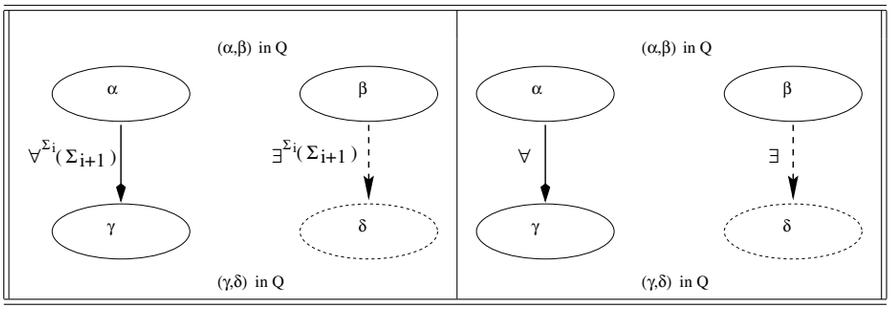

Fig. 5. The conditions in (2σ) on the quotient structures.

New_HHK$(\langle S, R_1, R_2\rangle, K, U)$

$P := K$;
for each $a \in S$ do
  $sim(a) := \{e \mid (a, e) \in K\}$;
  $rem(a) := S \setminus pre_1(sim(a))$;
while $\{a \mid rem(a) \neq \emptyset\} \neq \emptyset$ do
  let $a \in \{a \mid rem(a) \neq \emptyset\}$
  for each $b \in pre_2(a), c \in rem(a), c \in sim(b)$ do
    $sim(b) := sim(b) \setminus \{c\}$;
    $P := P \setminus \{(b, c)\}$;
    if U then for each $c_1 \in pre_1(c)$ do
      if $post_1(c_1) \cap sim(b) = \emptyset$ then $rem(b) := rem(b) \cup \{c_1\}$;
  $rem(a) := \emptyset$;
return P

Fig. 6. The **New_HHK** Function.

Lemma 2.

$$\textbf{New\_HHK}(D, K, \bot) = \tau_D(K) \quad \textit{and} \quad \textbf{New\_HHK}(D, K, \top) = Fix(\tau_D)(K).$$

Recalling that on the ground of Corollary 3 P_{i+1} can be computed from P_i as fix-point, the correctness of the procedures **Update** and **Refine** is based on:
Theorem 5. $\langle \Sigma_{i+1}, \mathsf{P}_{i+1}\rangle = \sigma(\langle \Sigma_i, \mathsf{P}_i\rangle)$.
As a consequence of Theorem 4 and of Corollary 1, since the **Stable Simulation Algorithm** terminates whenever $\Sigma_{i+1} = \Sigma_i$, we can conclude that it computes the solution $\langle M, \preceq\rangle$ of the GCPP over G and $\langle \Sigma, P\rangle$ performing at most $|M|$ iterations of the while-loop.

The complexity analysis is based on the following result:
Theorem 6. *Let $G = \langle N, E\rangle$ and $\langle \Sigma, P\rangle$ be a partition-pair with P transitive.* **Stable Simulation Algorithm** *computes the solution $\langle M, \preceq\rangle$ of the GCPP over them in time $O(|M|^2|E|)$ and in space $O(|M|^2 + |N|\log(|M|))$.*

6 Related Works

Recently the simulation relation has been algorithmically revisited in [4] and in [7]. Both works exploit a *partition refinement* argument in order to improve on the simulation algorithms in [13] and [3]. Given a labeled graph $G = \langle N, E, \Sigma \rangle$, the latter ([13,3]) can be used to obtain the simulation quotient over N in $O(|N||E|)$ time and $O(|N|^2)$ space. The main aim of the authors of [4] is that of keeping as low as possible the space requirements of their simulation algorithm. Indeed, despite its rather heavy time complexity ($O(|M|^4(|E| + |M|^2) + |M|^2|N|(|N| + |M|^2))$), the routine in [4] has a space complexity depending only on the size of the simulation quotient, $|M|$, given in output: $O(|M|^2 + |N| \log(|M|))$. As well as in [4], the space parameter is carefully thought of in the algorithm we have presented in this work: our major aim is that of designing a highly space-efficient procedure for deciding simulation equivalence that is not too costly in time. Our **Stable Simulation Algorithm**, as the one in [4], gives in output the simulation quotient together with the simulation partial order over its classes. Moreover it shares with [4] the same structure: each iteration consists of a step of partition's refinement followed by an update of a relation over such a partition. Indeed, the two algorithms have exactly the same space requirements but the procedure presented here has a better time complexity. While the **Partitioning Algorithm** in [4] need $O(|M|^2)$ iterations to get to a fix-point, our **Stable Simulation Algorithms** ends after $O(|M|)$ iterations only (cf. Theorem 6). Intuitively, on the ground of the characterization of the simulation given in Theorem 2, our algorithm updates "deeper" the relation over the partition so that in each refinement step at least one class *must* be split. Moreover, both the refinement and the update steps in the **Stable Simulation Algorithm** are less time-demanding than the corresponding functions in [4]. So, as stated in Theorem 6, the time complexity of our routine is $O(|M|^2|E|)$. Above we show an example in which our algorithm requires few iterations than the one in [4].

Example 1. Parts *a*), *b*) and *c*) of Figure 7 reflect, respectively, the partition-pairs on the depicted labeled graph after: *a*) the initialization phase of the **Stable Simulation Algorithm**; *b*) and *c*) the first and the second call to the subroutine **NewHHK** during the first call to the procedure **Update**. The **Stable Simulation Algorithm** needs only $O(1)$ iterations to get to the simulation quotient of the graph in 7, whereas the procedure in [4] needs $O(N)$ iterations. More in detail using [4], there are $O(N)$ iterations in which the partition never changes while the relation over its classes is successively refined.

The simulation algorithm in [7] also takes advantage of a *partition refinement* argument. However, while the partitions involved in [4] and in the **Stable Simulation Algorithm** are always coarser than the simulation quotient, the ones refined in [7] are always coarser than the bisimulation quotient. Given in input two labeled transition transition systems T_1 and T_2, the algorithm in [7] proceeds simultaneously minimizing via bisimulation the two graphs and determining the set of classes in T_2 able to simulate each class in T_1. The partition over a labeled

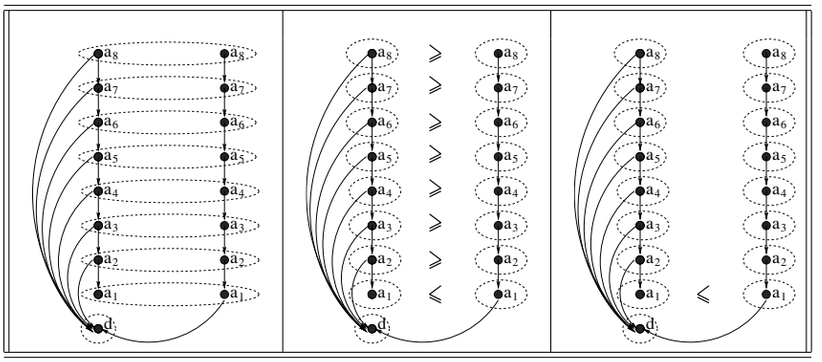

Fig. 7. Example of computation.

graph induced by the bisimulation equivalence is finer than the partition induced by the simulation: hence, a relation over its classes may have higher space requirements. However, the former equivalence can be computed faster than the latter because a "process-the-smallest-half" policy can be used (see [17,14]). Thus the algorithm proposed by Cleaveland and Tan in [7] achieves a time complexity which is today's state-of-the-art: $O(|B_1||B_2|+|E_1|\log(|N_1|)+|B_1||E_2|+|\varepsilon_1||B_2|)$, where ε_i and B_i represent the relation and the states'space of the transition system T_i reduced via bisimulation. Nevertheless, since at least on T_1 the entire bisimulation quotient must be computed, the deeper is the minimizing power of the simulation equivalence with respect to bisimulation, the less space-efficient the algorithm in [7] becomes (with respect to [4] as well as to our routine). For the sake of the argument, a family of examples on which [7] has higher space requirements than [4] as well than our procedure has the following features:

- T_1 is not reducible via bisimulation whereas the simulation equivalence has a deep minimizing power on it;
- T_2 follows T_1 in the simulation preorder over transition systems.

Figure 8 depicts two transition systems with the above features: whatever are the initial states of the two systems, T_2 simulates T_1. All the states belonging to the cycle in T_1 are simulation-equivalent and pairwise not bisimilar. In T_2 there are neither bisimilar nor similar states; hence, [7] require $O(|N|^2)$ space whereas the space requirement of the **Stable Simulation Algorithm** is bounded by $O(|N|\log(|N|))$.

We conclude by observing that, in a minimization framework, choosing the simulation equivalence rather than the bisimulation one is paying only if the simulation's power of reduction strongly overcomes bisimulation's one (because bisimulation is less time demanding than simulation). In other words simulation becomes worthwhile when $|M| << |B|$, i.e. when the routine in [7] and the one proposed here have comparable time complexities.

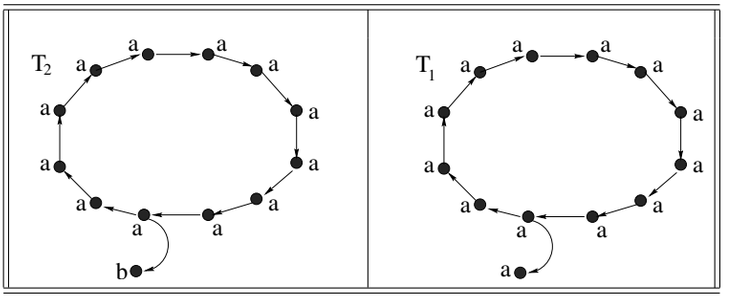

Fig. 8. Space Requirements.

7 Conclusions and Further Work

We think that the circle of ideas presented here can be useful in the study of fast simulation algorithm developed to operate in situations in which strong space constraint are present. Moreover, the use of the $\forall\exists$-structure and the definition of coarsest partition problems on them, seem to be a methodology with some potential in all those situations in which a fix-point in the lattice of equivalence relations is to be computed.

We plain to work on a symbolic implementation of our algorithm which is naturally suggested by the fact that our algorithm always works on sets of nodes (the classes of the partitions).

An attempt to combine negative and positive strategies to solve the coarsest partition problem presented here (as, in the case of bisimulation, has been done in [10]) is under study.

References

1. J. van Benthem. *Modal Correspondence Theory.* PhD thesis, Universiteit van Amsterdam, Instituut voor Logica en Grondslagenonderzoek van Exacte Wetenschappen, 1976.
2. B. Bloom. *Ready Simulation, Bisimulation, and the Semantics of CCS-like Languages.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1989.
3. Bard Bloom and Robert Paige. Transformational design and implementation of a new efficient solution to the ready simulation problem. *Science of Computer Programming*, 24(3):189–220, 1995.
4. D. Bustan and O. Grumberg. Simulation based minimization. In D.A. McAllester, editor, *Proc. 17th Int'l Conference on Automated Deduction (CADE'00)*, volume 1831 of *LNCS*, pages 255–270. Springer, 2000.
5. R. Cleaveland, J. Parrow, and B. Steffen. The Concurrency Workbench: A Semantics Based Tool for the Verification of Concurrent Systems. *ACM Transactions on Programming Languages and Systems*, 15(1):36–72, 1993.

6. R. Cleaveland and B. Steffen. A linear-time model-checking algorithm for the alternation-free modal mu-calculus. In K.G. Larsen and A. Skou, editors, *Proceedings of Computer Aided Verification (CAV'91)*, volume 575 of *LNCS*, pages 48–58. Springer, 1992.
7. R. Cleaveland and L. Tan. Simulation revised. In T. Margaria and W. Yi, editors, *Proc. 7th Int'l Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'01)*, volume 2031 of *LNCS*, pages 480–495. Springer, 2001.
8. D. Dams, R. Gerth, and O. Grumberg. Abstract interpretation of reactive systems. *ACM Transactions on Programming Languages and Systems*, 19(2):253–291, 1997.
9. N. Dershowitz and J.-P. Jouannaud. *Rewrite Systems*, volume B, chapter 6, pages 244–320. Elsevier/MIT press, 1990.
10. A. Dovier, C. Piazza, and A. Policriti. A fast bisimulation algorithm. In G. Berry, H. Comon, and A. Finkel, editors, *Proceedings of Computer Aided Verification (CAV'01)*, volume 2102 of *LNCS*, pages 79–90. Springer, 2001.
11. M. Forti and F. Honsell. Set theory with free construction principles. *Annali Scuola Normale Superiore di Pisa, Cl. Sc.*, IV(10):493–522, 1983.
12. R. Gentilini, C. Piazza, and A. Policriti. Simulation as coarsest partition problem. RR 04-01, Dep. of Computer Science, University of Udine, Italy, 2001. Available at http://www.dimi.uniud.it/~piazza/simul.ps.gz.
13. M. R. Henzinger, T. A. Henzinger, and P. W. Kopke. Computing simulations on finite and infinite graphs. In *36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, pages 453–462. IEEE Computer Society Press, 1995.
14. J.E. Hopcroft. An *nlogn* algorithm for minimizing states in a finite automaton. In *Theory of Machines and Computations, Ed. by Zvi Kohavi and Azaria Paz*, pages 189–196. Academic Press, 1971.
15. R. Milner. A calculus of communicating systems. In G. Goos and J. Hartmanis, editors, *Lecture Notes on Computer Science*, volume 92. Springer, 1980.
16. O. Grumberg and D.E. Long. Model checking and modular verification. *ACM Transactions on Programming Languages and systems*, 16(3):843–871, 1994.
17. R. Paige and R. E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, 1987.
18. R. Paige, R. E. Tarjan, and R. Bonic. A linear time solution to the single function coarsest partition problem. *Theoretical Computer Science*, 40(1):67–84, 1985.