Goal-Oriented Reduction of Automata Networks

Loïc Paulevé

Goal-Oriented Reduction of Automata Networks

Loïc Paulevé

CNRS, Laboratoire de Recherche en Informatique UMR CNRS 8623
Université Paris-Sud, 91405 Orsay Cedex, France
loic.pauleve@lri.fr

Abstract

We consider networks of finite-state machines having local transitions conditioned by the current state of other automata. In this paper, we depict a reduction procedure tailored for a given reachability property of the form "from global state s, there exists a sequence of transitions leading to a state where an automaton g is in a local state $\top$". By exploiting a causality analysis of the transitions within the individual automata, the proposed reduction removes local transitions while preserving all the minimal traces that satisfy the reachability property. The complexity of the procedure is polynomial in the total number of local states and transitions, and exponential in the number of local states within one automaton. Applied to automata networks modelling dynamics of biological systems, we observe that the reduction shrinks down significantly the reachable state space, enhancing the tractability of the model-checking of large networks.

1 Introduction

Automata networks model dynamical systems resulting from simple interactions between entities. Each entity is typically represented as an automaton with few internal states which evolve subject to the state of a narrow range of other entities in the network. Richness of emerging dynamics arises from several factors including the topology of the interactions, the presence of feedback loop, and the concurrency of transitions.

Automata networks are notably used to model biological systems, e.g., signalling networks which detail the successive, intertwined, and regulated transmission of a signal from the membrane of the cell toward the activation or inhibition of genes; and gene regulatory networks which model the thresholded up- and down-regulation circuits between genes. In such a context, it is crucial to confront the dynamics of a model to actual biological knowledge to validate or refute the model, and infer well-founded hypotheses on the important mechanisms controlling the emerging behaviours.

Part of those properties can be modelled as reachability properties, for instance for checking the (im)possibility of activation of an entity from a given set of initial states with a potential set of perturbations. Due to the increasing precision of biological knowledge, models of networks become larger and larger and can gather hundreds to thousands of interacting entities making the formal analysis of their dynamics a challenging task.

Facing a model too large for a raw exhaustive analysis, a natural approach is to reduce its dynamics while preserving important properties. Multiple approaches, generally complementary, have been explored since decades to address such a challenge in dynamical and concurrent systems [23, 11, 13]. Among those, structural reductions attempt to reduce the model specification while obtaining bisimulation, e.g., [22] on Petri nets; or preserving a subset of dynamical features and properties, e.g., the cone of influence reduction [4] which delimit the variables of the model having an influence of a given property, or Petri net transition agglomerations which preserve liveness and LTL properties [3, 8]. In the scope of rule-based models of biological networks, efficient static analysis methods have been developed to lump numerous global states of the systems based on the fragmentation of interacting components [7]; and to *a posteriori* compress simulated traces to obtain compact witnesses of dynamical properties [6]. Reduction methods focusing on the preservation of the dynamical attractors (long-term/steady-state behaviour) have also been studied for

chemical reaction networks [14] and Boolean networks [15]. The latter approach applies to formalisms close to automata networks but does not preserve reachability properties.

Contribution In this article, we introduce a reduction technique for automata networks which identifies transitions that do not contribute to a given reachability property; and hence can be ignored. The considered automata networks are finite sets of finite-state machines where transitions between their local states are conditioned by the state of other automata in the network. We use a general concurrent semantics where any number of automata can apply one transition within one step.

The reduction preserves all the minimal traces (in the sense that no step nor transition can be deleted) satisfying reachability properties of the form "from state s there exists successive steps that lead to a state where a given automaton g is in local state $g_\top$". The complexity of the procedure is polynomial in the total number of local states and transitions, and exponential in the size of one automaton. Therefore, the reduction is highly scalable for networks between multiple automata, where each have a few local states.

The identification of the transition set that include all the minimal traces is performed by a static analysis of the causality of local state changes within automata. Such an analysis extends previous static analysis of reachability properties by abstract interpretation [18, 17]. By exploiting the component decomposition of the transition system, and the explicit conditions for the transitions between the local states, necessary or sufficient condition for reachability can be derived, and can be verified very efficiently. The work presented here widen the impact of such a static analysis of causality to produce reduced model of automata networks that conserve important dynamical features.

The effectiveness of our goal-oriented reduction is experimented on actual models of biological networks and show significant shrinkage of the dynamics of the automata networks, enhancing the tractability of a concrete verification. Compared to other existing reduction techniques, our goal is somehow close to the cone of influence reduction mentioned above, which identifies variables that do not impact a given property, except that our approach offers a much fine grained analysis in order to identify the sufficient transitions and values of variables that contribute to the property.

Outline The paper is structured as follows. Section 2 sets up the definition and semantics of the automata networks considered in this paper, together with the local causality analysis for reachability properties, based on prior work. Section 3 first depicts a necessary condition using local causality analysis for satisfying a reachability property and then introduce the goal-oriented reduction with the proof of minimal traces conservation. Section 4 shows the efficiency of such a reduction as a pre-processing step for model-checking of automata networks derived from biological networks. Finally, section 5 discusses the results and motivate further work.

Notations Integer ranges are noted $[m; n] = \{m, m+1, \cdots, n\}$. Given a finite set A, $\#A$ is the cardinality of A; $\wp(A)$ is the power set of A; $\subset$ denotes the subset inclusion (possibly equal). Given $n \in \mathbb{N}$, $x = (x^i)_{i\in[1;n]}$ is a sequence of elements indexed by $i \in [1; n]$; $\#x = n$; $x^{m..n}$ is the subsequence $(x^i)_{i\in[m;n]}$; ε is the empty sequence. $\wedge$ and $\vee$ are the usual logical *and* and *or* connectors.

2 Automata Networks and Local Causality

The definition and semantics of automata networks is presented in section 2.1. The local causality analysis for reachability properties on which relies the reduction is defined in section 2.2.

2.1 Automata Networks

We consider an Automata Network (AN) as a finite set of finite-state machines having transitions between their local states conditioned by the state of other automata in the network. An Automata Network is defined by a triple (Σ, S, T) (definition 1) where Σ is the set of automata identifiers; S associates to each automaton a finite set of local states: if $a \in \Sigma$, $S(a)$ refers to the set of local states of a; and T is the list of transitions between local states within each automaton. Each local state is written of the form a_i,

where $a \in \Sigma$ is the automaton in which the state belongs to, and i is a unique identifier; therefore given $a_i, a_j \in S(a)$, $a_i = a_j$ if and only if a_i and a_j refers to the same local state of the automaton a. For each automaton $a \in \Sigma$, $T(a)$ refers to the set of transitions of the form $t = a_i \xrightarrow{\ell} a_j$ with $a_i, a_j \in S(a)$, $a_i \neq a_j$, and ℓ is a set (possibly empty) of local states of automata other than a. The *pre-condition* of transition t, noted ${}^\bullet t$, is the set composed of a_i and of the local states in ℓ; the *post-condition*, noted $t^\bullet$ is the set composed of a_j and of the local states in ℓ.

Definition 1 (Automata Network (Σ, S, T))**.** An *Automata Network* (AN) is defined by a tuple (Σ, S, T) where

- Σ is the finite set of automata identifiers;
- For each $a \in \Sigma$, $S(a) = \{a_i, \ldots, a_j\}$ is the finite set of local states of automaton a; $S = \prod_{a \in \Sigma} S(a)$ is the finite set of global states;
  $\mathbf{LS} = \bigcup_{a \in \Sigma} S(a)$ denotes the set of all the local states.
- $T = \{a \mapsto T_a \mid a \in \Sigma\}$, where $\forall a \in \Sigma, T_a \subset S(a) \times \wp(\mathbf{LS} \setminus S(a)) \times S(a)$ with $(a_i, \ell, a_j) \in T_a \Rightarrow a_i \neq a_j$, is the mapping from automata to their finite set of local transitions.

We note $a_i \xrightarrow{\ell} a_j \in T \overset{\Delta}{\Leftrightarrow} (a_i, \ell, a_j) \in T(a)$ and $a_i \to a_j \in T \overset{\Delta}{\Leftrightarrow} \exists \ell \in \wp(\mathbf{LS} \setminus S(a)), a_i \xrightarrow{\ell} a_j \in T$. Given $t = a_i \xrightarrow{\ell} a_j \in T$, $\mathrm{orig}(t) \overset{\Delta}{=} a_i$, $\mathrm{dest}(t) \overset{\Delta}{=} a_j$, $\Gamma(t) \overset{\Delta}{=} \ell$, ${}^\bullet t \overset{\Delta}{=} \{a_i\} \cup \ell$, and $t^\bullet \overset{\Delta}{=} \{a_j\} \cup \ell$.

At any time, each automaton is in one and only one local state, forming the global state of the network. Assuming an arbitrary ordering between automata identifiers, the set of global states of the network is referred to as S as a shortcut for $\prod_{a \in \Sigma} S(a)$. Given a global state $s \in S$, $s(a)$ is the local state of automaton a in s, i.e., the a-th coordinate of s. Moreover we write $a_i \in s \overset{\Delta}{\Leftrightarrow} s(a) = a_i$; and for any $ls \in \wp(\mathbf{LS})$, $ls \subset s \overset{\Delta}{\Leftrightarrow} \forall a_i \in s, s(a) = a_i$.

In the scope of this paper, we allow, but do not enforce, the parallel application of transitions in different automata. This lead to the definition of a *step* as a set of transitions, with at most one transition per automaton (definition 2). Note that for notational convenience in the next section, we allow empty steps. The pre-condition (resp. post-condition) of a step τ, noted ${}^\bullet\tau$ (resp. $\tau^\bullet$), naturally extends to the union of the pre-conditions (resp. post-conditions) of compositing transitions. A step τ is *playable* in a state $s \in S$ if and only if ${}^\bullet\tau \subset s$, i.e., all the local states in the pre-conditions of transitions are in s. If τ is playable in s, $s \cdot \tau$ denotes the state after the applications of all the transitions in τ, i.e., where for each transition $a_i \xrightarrow{\ell} a_j \in \tau$, the local state of automaton a has been replaced with a_j. Remark that $\tau^\bullet \subset s \cdot \tau$ and that this definition implicitly rules out steps composed of incompatible or ill-defined transitions, where different local states of a same automaton are in the pre-condition.

Definition 2 (Step)**.** Given an AN (Σ, S, T), a *step* τ is a sub-set of local transitions T such that for each automaton $a \in \Sigma$, there is at most one local transition $T(a)$ in τ ($\forall a \in \Sigma, \#(\tau \cap T(a)) \leq 1$).
We note ${}^\bullet\tau \overset{\Delta}{=} \bigcup_{t \in \tau} {}^\bullet t$ and $\tau^\bullet \overset{\Delta}{=} \bigcup_{t \in \tau} t^\bullet$. Given a state $s \in S$ such that ${}^\bullet t \subset s$, $s \cdot \tau$ denotes the state s where local states in ${}^\bullet\tau$ have been replaced with $\tau^\bullet$: $\forall a \in \Sigma$, $(s \cdot \tau)(a) = a_j$ if $\exists a_i \to a_j \in \tau$, and $(s \cdot \tau)(a) = s(a)$ otherwise.

A sequence of steps that are successively playable in a state $s \in S$ is called a *trace* (definition 3). The pre-condition ${}^\bullet\pi$ of a trace π is the set of local states that have to be in a state s in order to apply π (${}^\bullet\pi \subset s$); and the post-condition $\pi^\bullet$ is the set of local states that are present in the state after the full application of a trace ($\pi^\bullet \subset s \cdot \pi$).

Definition 3 (Trace)**.** Given an AN (Σ, S, T) and a state $s \in S$, a *trace* π is a sequence of steps such that $\forall i \in [1; \#\pi]$, ${}^\bullet\pi^i \subset (s \cdot \pi^1 \cdot \cdots \cdot \pi^{i-1})$.
The pre-condition ${}^\bullet\pi$ and the post-condition $\pi^\bullet$ are defined as follows: for all $n \in [1; \#\pi]$, for all $a_i \in {}^\bullet\pi^n$, $a_i \in {}^\bullet\pi \overset{\Delta}{\Leftrightarrow} \forall m \in [1; n-1], S(a) \cap {}^\bullet\pi^m = \emptyset$; similarly, for all $n \in [1; \#\pi]$, for all $a_j \in \pi^{n\bullet}$, $a_j \in \pi^\bullet \overset{\Delta}{\Leftrightarrow} \forall m \in [n+1; m], S(a) \cap \pi^{m\bullet} = \emptyset$. If π is empty, ${}^\bullet\pi = \pi^\bullet = \emptyset$.
The set of transitions composing a trace π is noted $\mathrm{tr}(\pi) \overset{\Delta}{=} \bigcup_{n=1}^{\#\pi} \pi^n$.

Given an Automata Network (Σ, S, T) and a state $s \in S$, the local state $g_\top \in \mathbf{LS}$ is *reachable* from s if and only if either $g_\top \in s$ or there exists a trace π with ${}^\bullet\pi \subset s$ and $g_\top \in \pi^\bullet$. We consider a trace π for $g_\top$ reachability from s is minimal if and only if there exists no other trace reaching $g_\top$ with a strict sub-sequence of steps, or with strict subset of steps of π (definition 4). Equivalently, a trace is minimal for $g_\top$ reachability if no steps or transitions can be removed from it without breaking the trace validity or $g_\top$ reachability.

Definition 4 (Minimal trace for local state reachability)**.** A trace π is *minimal* w.r.t. $g_\top$ reachability from s if and only if there is no trace ϖ, $\varpi \neq \pi$, $\#\varpi \leq \#\pi$, $g_\top \in \varpi^\bullet$, such that there exists an injection $\phi : [1; \#\varpi] \to [1; \#\pi]$ with $\forall i, j \in [1; \#\varpi]$, $i < j \Leftrightarrow \phi(i) < \phi(j)$ and $\varpi^i \subseteq \pi^{\phi(i)}$.

Automata networks as presented can be considered as a class of 1-safe Petri Nets [2] (at most one token per place) having groups of mutually exclusive places, acting as the automata, and where each transition has one and only one in-going and out-going arc and any number of read arcs. The semantics considered in this paper where transitions within different automata can be applied simultaneously echoes with Petri net step-semantics and concurrent/maximally concurrent semantics [10, 19]. In the Boolean network community, such a semantics is referred to as the asynchronous generalized update schedule [1].

2.2 Local Causality

Locally reasoning within one automaton a, we call the reachability of one of its local state a_j from s with $s(a) = a_i$ an *objective*, noted $a_i \to^* a_j$ (definition 5).

Definition 5 (Objective)**.** Given an Automata Network (Σ, S, T), the reachability of local state a_j from a_i is called an *objective* and is denoted $a_i \to^* a_j$. The set of all objectives is referred to as $\mathbf{Obj} \triangleq \{a_i \to^* a_j \mid (a_i, a_j) \in \mathbf{LS} \times \mathbf{LS}\}$.

Given an objective $a_i \to^* a_j \in \mathbf{Obj}$, local-paths$(a_i \to^* a_j)$ (definition 6) is the set of all the acyclic local paths of transitions $T(a)$ within automaton a from a_i to a_j. A local path is not necessarily a trace, as transitions may be conditioned by the state of other automata in the network; however, any trace reaching first a_i and then a_j uses all the transitions of one local path in local-paths$(a_i \to^* a_j)$, as stated by property 1. One can remark that the number of local acyclic paths within an automaton a is polynomial in the number of transitions $T(a)$ and exponential in the number of local states in a.

Definition 6 (local-paths)**.** For each objective $a_i \to^* a_j \in \mathbf{Obj}$, if $i = j$, local-paths$(a_i \to^* a_i) \triangleq \{\varepsilon\}$; if $i \neq j$, a sequence η of transitions in $T(a)$ is in local-paths$(a_i \to^* a_j)$ if and only if $\#\eta \geq 1$, $\text{orig}(\eta^1) = a_i$, $\text{dest}(\eta^{\#\eta}) = a_j$, $\forall n \in [1; \#\eta - 1]$, $\text{dest}(\eta^n) = \text{orig}(\eta^{n+1})$, and $\forall n, m \in [1; \#\eta], n \neq m \Rightarrow \text{orig}(\eta^n) \neq \text{dest}(\eta^m)$.

Property 1. For any trace π, for all $a \in \Sigma$, for all $a_i, a_j \in S(a)$, for all $n, m \in [1; \#\pi]$, $a_i \in {}^\bullet\pi^n$ and $a_j \in \pi^{m\bullet}$ only if $\exists \eta \in$ local-paths$(a_i \to^* a_j)$ such that exists an injection $\phi : [1; \#\eta] \to [n; m]$ with $\forall u, v \in [1; \#\eta], u < v \Leftrightarrow \phi(u) < \phi(v)$ and $\eta^u \in \pi^{\phi(u)}$.

Example 1. Let us consider the automata network (Σ, S, T), graphically represented in figure 1, where:

$$\Sigma = \{a, b, c, c\}$$

$$S(a) = \{a_0, a_1\} \qquad T(a) = \{a_0 \xrightarrow{\{b_0\}} a_1, a_1 \xrightarrow{\emptyset} a_0\}$$

$$S(b) = \{b_0, b_1\} \qquad T(b) = \{b_0 \xrightarrow{\{a_1\}} b_1, b_1 \xrightarrow{\{a_0\}} b_0\}$$

$$S(c) = \{c_0, c_1, c_2\} \qquad T(c) = \{c_0 \xrightarrow{\{a_1\}} c_1, c_1 \xrightarrow{\{b_1\}} c_0, c_1 \xrightarrow{\{b_0\}} c_2, c_0 \xrightarrow{\{d_1\}} c_2\}$$

$$S(d) = \{d_0, d_1\} \qquad T(d) = \emptyset$$

The local paths for the objective $c_0 \to^* c_2$ are local-paths$(c_0 \to^* c_2) = \{c_0 \xrightarrow{\{a_1\}} c_1 \xrightarrow{\{b_0\}} c_2, c_0 \xrightarrow{\{d_1\}} c_2\}$.
From the state $\langle a_0, b_0, c_0, d_0\rangle$, instances of traces are
$\{a_0 \xrightarrow{\{b_0\}} a_1\}, \{b_0 \xrightarrow{\{a_1\}} b_1, c_0 \xrightarrow{\{a_1\}} c_1\}, \{a_1 \xrightarrow{\emptyset} a_0\}, \{b_1 \xrightarrow{\{a_0\}} b_0\}, \{c_1 \xrightarrow{\{b_0\}} c_2\}$;
$\{a_0 \xrightarrow{\{b_0\}} a_1\}, \{c_0 \xrightarrow{\{a_1\}} c_1\}, \{c_1 \xrightarrow{\{b_0\}} c_2\}$;
the latter only being a minimal trace for c_2 reachability.

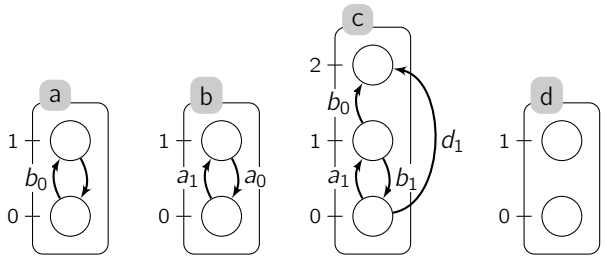

Figure 1: An example of Automata Network. Automata are represented by labelled boxes, and local states by circles where ticks are their identifier within the automaton – for instance, the local state a_0 is circle ticked 0 in the box a. A transition is a directed edge between two local states within the same automaton. It can be labelled with a set of local states of other automata. In this example, all the transitions are conditioned by at most one other local state.

3 Goal-Oriented Reduction

Assuming a global Automata Network (Σ, S, T), an initial state $s \in S$ and a reachability goal $g_\top$ where $g \in \Sigma$ and $g_\top \in S(g)$, the goal-oriented reduction identifies a subset of local transitions T that are sufficient for reproducing all the minimal traces leading to $g_\top$ from s. The reduction procedure takes advantage of the local causality analysis both to fetch the transitions that matters for the reachability goal and to filter out objectives that can be proven impossible.

3.1 Necessary condition for local reachability

Given an objective $a_i \to^* a_j$ and a global state $s \in S$ where $s(a) = a_i$, prior work have demonstrated necessary conditions that need to be satisfied by the network for the existence of a trace leading to a_j from s [18, 17]. Those necessary conditions rely on the local causality analysis defined in previous section which extract necessary steps that have to be performed in order to reach the concerned local state.

Several necessary conditions have been demonstrated in [18] that take into account several informations extracted from local paths (dependencies, sequentiality, partial order constraints, ...). The complexity of deciding most of these necessary conditions is polynomial in the total number of local states and exponential in the number of local states within one automaton.

In the remaining of this section, we consider a generic predicate $\mathbf{valid}_s$ which is false only when an applied to an objective such that there exists no trace concretizing it from s.

Definition 7 ($\mathbf{valid}_s$)**.** Given any objective $P \in \mathbf{Obj}$, $\mathbf{valid}_s(P)$ if there exists a trace π from s such that $\exists m, n \in [1; \#\pi]$ with $m \leq n$, $a_i \in {}^\bullet\pi^m$, and $a_j \in \pi^{n\bullet}$.

For the sake of self-consistency, we give in proposition 1 an implementation of such a predicate which is a simplified version of a necessary condition demonstrated in [18]. Essentially, an objective is valid if there exists a local path for the objective where all the objectives from the initial state to any of the condition of the compound transitions are valid. Such a set of valid objective can be computed iteratively, starting from objectives of the form $a_i \to^* a_i$, and progressively add objectives having a local path where conditions lead to already validated objectives.

Proposition 1. For all objective $P \in \mathbf{Obj}$, $\mathbf{valid}_s(P) \overset{\Delta}{\Leftrightarrow} P \in \Omega$ where Ω is the least fixed point of the monotonic function $\mathrm{F} : \wp(\mathbf{Obj}) \to \wp(\mathbf{Obj})$, with

$$\mathrm{F}(\Omega) \overset{\Delta}{=} \{a_i \to^* a_j \in \mathbf{Obj} \mid \exists \eta \in \text{local-paths}(a_i \to^* a_j) : \forall n \in [1; \#\eta], \forall b_k \in \Gamma(\eta^n), s(b) \to^* b_k \in \Omega\} \ .$$

Applied to the Automata Network of figure 1, if $s = \langle a_0, b_0, c_0, d_0 \rangle$, $\mathbf{valid}_s(c_0 \to^* c_2)$ because $c_0 \xrightarrow{a_1} c_1 \xrightarrow{b_0} c_2 \in \text{local-paths}(c_0 \to^* c_2)$ with $\mathbf{valid}_s(a_0 \to^* a_1)$ and $\mathbf{valid}_s(b_0 \to^* b_0)$; but not $\mathbf{valid}_s(d_0 \to^* d_1)$.

More restrictive implementations of $\mathbf{valid}_s$ may be defined, for instance following [18], which is considered out of the scope of this paper.

3.2 Reduction procedure

With this section, we introduce the goal-oriented reduction procedure which aims at identifying transitions that do not take part in any minimal trace leading from the given initial state to the goal local state. Essentially, the goal-oriented reduction exploits the local causality analysis to focus only on objectives involved in the goal reachability while redacting objectives that do not satisfy necessary condition for concretizability.

The reduction procedure we propose consists in collecting a set $\mathcal{B}$ of objectives that may contribute to a minimal trace for the goal reachability. The construction of $\mathcal{B}$ from s and $g_\top$ is detailed in definition 8. To ease notations, and without loss of generalization, we assume that any automaton a is in state a_0 in s. Initially starting with the main objective $g_0 \to^* g_\top$, the procedure iteratively collects objectives that may be involved in the undergoing local paths of already collected objectives. The considered local paths are only those where all transition conditions rely on valid objectives (definition 7). The transitions composing such local paths are noted $\mathrm{tr}(\mathcal{B})$. The involved objectives are identified as follows: if a transition $b_j \xrightarrow{\ell} b_k$ is in $\mathrm{tr}(\mathcal{B})$, for each transition condition $a_i \in \ell$, the objective $a_0 \to^* a_i$ is added in $\mathcal{B}$, and for each other objective $b_\star \to^* b_i \in \mathcal{B}$ of automaton b, the objective $b_k \to^* b_i$ is added in $\mathcal{B}$. Whereas the first criteria references the objectives required to concretize a local path, the second criteria accounts for the possible interleaving and successions of objectives for a same automaton.

Definition 8 ($\mathcal{B}$)**.** Given an Automata Network (Σ, S, T), an initial state s where, without loss of generality, $\forall a \in \Sigma$, $s(a) = a_0$, and a local state $g_\top$ with $g \in \Sigma$ and $g_\top \in S(g)$, $\mathcal{B} \subset \mathbf{Obj}$ is the smallest set which satisfies the following conditions:

1. $g_0 \to^* g_\top \in \mathcal{B}$
2. $b_j \xrightarrow{\ell} b_k \in \mathrm{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \to^* a_i \in \mathcal{B}$
3. $b_j \xrightarrow{\ell} b_k \in \mathrm{tr}(\mathcal{B}) \Rightarrow \forall b_\star \to^* b_i \in \mathcal{B}, b_k \to^* b_i \in \mathcal{B}$

with

$$\mathrm{tr}(\mathcal{B}) \triangleq \bigcup_{P \in \mathcal{B}} \mathrm{tr}(\text{local-paths}_s(P)) \ ,$$

where, $\forall P \in \mathbf{Obj}$,

$$\text{local-paths}_s(P) \triangleq \{\eta \in \text{local-paths}(P) \mid \forall n \in [1; \#\eta], \forall b_k \in \Gamma(\eta^n), \mathbf{valid}_s(b_0 \to^* b_k)\} \ .$$

Theorem 1 states the main property emerging from definition 8: any trace which is minimal for the reachability of $g_\top$ from initial state s is composed only of transitions in $\mathrm{tr}(\mathcal{B})$. The proof is given in section 3.3. Therefore, the Automata Network $(\Sigma, S, \mathrm{tr}(\mathcal{B}))$ contains less transitions but preserves all the minimal traces for the goal reachability.

Theorem 1. *For each* minimal *trace* π *reaching* $g_\top$ *from* s, $\mathrm{tr}(\pi) \subset \mathrm{tr}(\mathcal{B})$.

The construction of $\mathcal{B}$ is done by a progressive exploration of related objectives and local paths. Noting that there is at most $\sum_{a \in \Sigma} \#S(a)^2$ objectives, and that the number of acyclic local paths is exponential in the number of local states in the automaton, the overall complexity of the reduction, including the necessary condition checking in proposition 1, is polynomial in the total number of local states and transitions, and exponential in the size of one automaton.

Figure 2 shows the results of the reduction of the example Automata Network of figure 1 for the reachability of c_2 from the state where all automata start at 0. Basically, the local path from c_0 to c_2 using d_1 being impossible to concretize (because $\mathbf{valid}_s(d_0 \to^* d_1)$ is false), it has been removed, and consequently, so are the transitions involving b_1 as b_1 is not required for c_2 reachability.

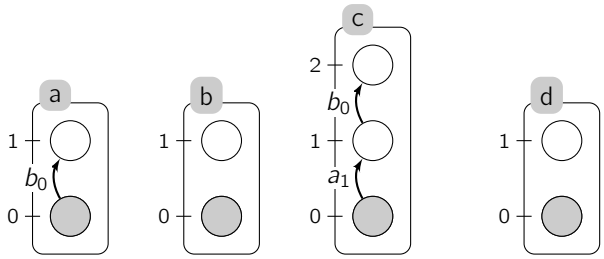

Figure 2: Reduced Automata Network from figure 1 for the reachability of c_2 from initial state indicated in grey.

3.3 Proof of minimal traces conservation

Let us consider any trace π that is minimal for $g_\top$ reachability from $s \in S$.

From property 1 and definition 7, any trace reaching first a_i and then a_j uses all the transitions of one local path in local-paths$_s(a_i \to^* a_j)$.

We first prove with lemma 2 that the last transition of π, of the form $\pi^{\#\pi} = \{g_i \to g_\top\}$, is necessarily in tr($\mathcal{B}$). Indeed, by definition of $\mathcal{B}$, $g_0 \to^* g_\top \in \mathcal{B}$; therefore by lemma 1, $g_i \to g_\top \notin$ local-paths$_s(g_0 \to^* g_\top)$ implies that reaching g_i requires to reach $g_\top$ beforehand.

Lemma 1. *If $a_j \to a_i \notin$ tr(local-paths$_s(a_0 \to^* a_i)$), for any trace π from s such that $\exists v \in [1; \#\pi]$ with $a_j \in \pi^{v\bullet}$, there exists $u \in [1; v-1]$ such that $a_i \in \pi^{u\bullet}$.*

Proof. If there exists an acyclic sequence $\eta \in$ local-paths$_s(a_0 \to^* a_j)$ where a_i does not belong to, the sequence $\eta :: a_j \to a_i$ is acyclic and, by definition, belongs to local-paths$_s(a_0 \to^* a_i)$, which is a contradiction. □

Lemma 2. *If π is a minimal trace for $g_\top$ reachability from state s, then, necessarily, $\pi^{\#\pi} \subset$ tr($\mathcal{B}$).*

Proof. As π is minimal for $g_\top$ reachability, without loss of generality, we can assume that $\pi^{\#\pi} = \{g_i \to g_\top\}$. By definition, tr(local-paths$_s(g_0 \to^* g_\top)$) $\subset$ tr($\mathcal{B}$). By lemma 1, if $g_i \to g_\top \notin$ tr(local-paths$_s(g_0 \to^* g_\top)$), then there exists $u < \#\pi$ such that $g_\top \in \pi^{u\bullet}$; hence, π would be non minimal. □

The rest of the proof of theorem 1 can be derived by contradiction: the main idea is to demonstrate that if a transition in π is not in tr($\mathcal{B}$), we can remove a set of transitions from π while preserving $g_\top$ reachability; therefore π is not minimal.

Given a transition $a_i \to a_j$ in the q-th step of π that is not in tr($\mathcal{B}$), removing $a_i \to a_j$ from π^q would imply to remove any further transition that is causally dependent on a_j. Two cases arise from this fact: either all further transitions that depends on a_j must be removed; or $a_i \to a_j$ is part of loop, and it is sufficient to remove the loop from π.

Lemma 3 ensures that if an objective $a_z \to^* a_k$ is in $\mathcal{B}$ and if a_z occurs before the q-th step and a_k after the q-th step, then $a_i \to a_j \notin$ tr(local-paths$_s(a_z \to^* a_k)$) only if $a_i \to a_j$ is part of a loop, i.e., there are two steps surrounding q where the automaton a is in the same state after their application.

Lemma 3. *Given $q \in [1; \#\pi]$ and $a \in \Sigma$ such that $\exists a_i \to a_j \in \pi^q$ with $a_i \to a_j \notin$ tr($\mathcal{B}$) and $\exists u, v \in [1; \#\pi]$ where $u \leq q < v$, $a_z \in {}^\bullet\pi^u$, $a_k \in \pi^v$, with $a_z \to^* a_k \in \mathcal{B}$, then there exists $m, n \in [u; v]$ with $m \leq q \leq n$ such that $(\pi^{1..m-1})^\bullet \cap S(a) = (\pi^{1..n})^\bullet \cap S(a)$. Moreover, if $a_k \in {}^\bullet\pi^v$, $n < v$.*

Proof. As $a_i \to a_j \notin$ tr($\mathcal{B}$) and tr(local-paths$_s(a_z \to^* a_k)$) $\subset$ tr($\mathcal{B}$), it results that $a_i \to a_j \notin$ tr(local-paths$_s(a_z \to^* a_k)$). Therefore $a_i \to a_j$ belongs to a loop of a local path from a_z (at index u in π) to a_k (at index v in π). Hence, $\exists m, n \in [u; v]$ with $m \leq q \leq n$ and $a_h, a_x, a_y \in S(a)$ such that $a_h \to a_x \in \pi^m$ and $a_y \to a_h \in \pi^n$; therefore $(\pi^{1..m-1})^\bullet \cap S(a) = (\pi^{1..n})^\bullet \cap S(a) = a_h$. In the case where $a_k \in {}^\bullet\pi^v$, $a_k \neq a_h$, hence $n < v$. □

Redacting such a loop from π may, again, enforce to remove other transitions from π that depend on those transitions occurring during the loop. Lemma 4 establishes that we can always delimit a loop of automaton a such that any transition that depend on the automaton a during the loop is not in tr($\mathcal{B}$).

Basically, such a property derives from the fact that if a transition in $\mathrm{tr}(\mathcal{B})$ depends on a local state of a, let us call it a_p, the objectives $a_0 \rightarrow^* a_p$ and $a_p \rightarrow^* a_k$ are in $\mathcal{B}$: lemma 3 can then be applied on the subpart of π that contains the transition $a_i \rightarrow a_j$ not in $\mathrm{tr}(\mathcal{B})$ and that concretizes either $a_0 \rightarrow^* a_p$ or $a_p \rightarrow^* a_k$ to identify a smaller loop containing $a_i \rightarrow a_j$.

Lemma 4. *Given $q \in [1; \#\pi]$ and $a \in \Sigma$ with $a_i \rightarrow a_j \in \pi^q$ but $a_i \rightarrow a_j \notin \mathrm{tr}(\mathcal{B})$, there exists $m, n \in [1; \#\pi]$ with $m \leq q \leq n$ such that $\forall t \in \mathrm{tr}(\pi^{m+1..n})$, $\Gamma(t) \cap S(a) \neq \emptyset \implies t \notin \mathrm{tr}(\mathcal{B})$. Moreover, if there exists $v \in [n+1; \#\pi]$ such that $\exists t \in \mathrm{tr}(\pi^v) \cap \mathrm{tr}(\mathcal{B})$ with $\Gamma(t) \cap S(a) \neq \emptyset$, then $(\pi^{1..m-1})^\bullet \cap S(a) = (\pi^{1..n})^\bullet \cap S(a)$; otherwise if $a = g$, then $(\pi^{1..m-1})^\bullet \cap S(a) = (\pi^{1..n})^\bullet \cap S(a)$ with $n < \#\pi$.*

Proof. First, let us assume that $a \neq g$ and for any $t \in \pi^{q+1..\#\pi}$, $\Gamma(t) \cap S(a) \neq \emptyset \Rightarrow t \notin \mathrm{tr}(\mathcal{B})$: the lemma is verified with $m = q$ and $n = \#\pi$.

Then, let us assume there exists $v \in [q+1; \#\pi]$ such that $\exists t \in \mathrm{tr}(\pi^v) \cap \mathrm{tr}(\mathcal{B})$ with $a_k \in \Gamma(t)$. By definition 8, this implies $a_0 \rightarrow^* a_k \in \mathcal{B}$. By lemma 3, there exists $m, n \in [1; v-1]$ with $m \leq q \leq n$ such that $(\pi^{1..m-1})^\bullet \cap S(a) = (\pi^{1..n})^\bullet \cap S(a)$.

Otherwise, $a = g$, and by lemma 3 with $a_k = g_\top$, there exists $m, n \in [1; \#\pi]$ with $m \leq q \leq n$ and $m \neq n$ such that $(\pi^{1..m-1})^\bullet \cap S(a) = (\pi^{1..n})^\bullet \cap S(a)$. Remark that it is necessary that $n < \#\pi$: if $n = \#\pi$, $g_\top \in (\pi^{1..m-1})^\bullet$, so π would be not minimal.

In both cases, if there exists $r \in [m+1; n]$ such that $\exists a_p \in S(a)$ and $\exists t \in \pi^r$ with $a_p \in \Gamma(t)$, then $t \in \mathrm{tr}(\mathcal{B})$ implies that $a_0 \rightarrow^* a_p \in \mathcal{B}$ and $a_p \rightarrow^* a_k \in \mathcal{B}$ (definition 8). If $r > q$, by lemma 3 with $a_k = a_p$ and $v = r$, there exists $m', n' \in [m+1; n]$ such that $m' \leq q \leq n' < r \leq n$ with $(\pi^{1..m'-1})^\bullet \cap S(a) = (\pi^{1..n'})^\bullet \cap S(a)$. If $r \leq q$, by lemma 3 with $a_0 = a_p$ and $u = r$, there exists $m', n' \in [m+1; n]$ such that $r \leq m' \leq q \leq n'$ with $(\pi^{1..m'-1})^\bullet \cap S(a) = (\pi^{1..n'})^\bullet \cap S(a)$. Therefore, by induction with lemma 3, there exists $m, n \in [1; \#\pi]$ such that $\forall t \in \mathrm{tr}(\pi^{m+1..n})$, $\Gamma(t) \cap S(a) \neq \emptyset \Rightarrow t \notin \mathrm{tr}(\mathcal{B})$. □

Given $a \in \Sigma$ and $q \in [1; \#\pi]$ such that $\exists t \in \pi^q$ with $\Sigma(t) = a$ and $t \notin \mathrm{tr}(\mathcal{B})$, let us refer to the couple $m, n \in [1; \#\pi]$ from lemma 4 with $\mathrm{cb}(\pi, a, q)$: $\mathrm{cb}(\pi, a, q) = (m, n) \Rightarrow \forall t \in \mathrm{tr}(\pi^{m+1..n}), \Gamma(t) \cap S(a) = \emptyset \vee t \notin \mathrm{tr}(\mathcal{B})$, and $n \neq \#\pi \Rightarrow (\pi^{1..m-1})^\bullet \cap S(a) = (\pi^{1..n})^\bullet \cap S(a)$.

We use lemma 4 to collect the portions of π to redact according to each automaton. We start from the last transition in π that is not in $\mathrm{tr}(\mathcal{B})$: if $\mathrm{tr}(\pi) \not\subset \mathrm{tr}(\mathcal{B})$, there exists $l \in [1; \#\pi]$ such that $\pi^l \not\subset \mathrm{tr}(\mathcal{B})$ and $\forall n > l, \pi^n \subset \mathrm{tr}(\mathcal{B})$. By lemma 2, we know that $l < \#\pi$. Let us denote by $b_i \rightarrow b_j$ one of the transitions in π^l which is not in $\mathrm{tr}(\mathcal{B})$. Let us define $\Psi \subset \Sigma \times [1; \#\pi] \times [1; \#\pi]$ the smallest set which satisfies:

- $(b, m, n) \in \Psi$ if $\mathrm{cb}(\pi, l, b) = (m, n)$
- $\forall (a, m, n) \in \Psi$, $\forall q \in [m+1; n]$, $\forall t \in \pi^q$, $\Gamma(t) \cap S(a) \neq \emptyset \implies (\Sigma(t), m', n') \in \Psi$ where $\mathrm{cb}(\pi, q, \Sigma(t)) = (m', n')$.

Finally, let us define the sequence of steps ϖ as the sequence of steps π where the portion of transitions referenced in Ψ are removed: for each step $q \in [1; \#\pi]$, the transition $a_i \rightarrow a_j$ is removed if and only if there exists $(a, m, n) \in \Psi$ with $m \leq q \leq n$. Formally, $\#\varpi = \#\pi$ and for all $q \in [1; \#\pi]$, $\varpi^q \triangleq \{t \in \pi^q \mid \nexists (a, m, n) \in \Psi : a = \Sigma(t) \wedge m \leq q \leq n\}$.

From lemma 4 and Ψ definition, ϖ is a valid trace. Moreover, by lemma 4, there is no $q \in [1; \#\pi]$ such that $(g, q, \#\pi) \in \Psi$, hence $g_\top \in \varpi^\bullet$. Therefore, π is not minimal, which contradicts our hypothesis.

For instance, consider the reachability of c_2 in the Automata Network of figure 1 from state $\langle a_0, b_0, c_0, d_0 \rangle$ with the reduced Automata Network in figure 2. Given the trace
$\pi = \{a_0 \xrightarrow{\{b_0\}} a_1\}, \{b_0 \xrightarrow{\{a_1\}} b_1, c_0 \xrightarrow{\{a_1\}} c_1\}, \{a_1 \xrightarrow{\emptyset} a_0\}, \{b_1 \xrightarrow{\{a_0\}} b_0\}, \{c_1 \xrightarrow{\{b_0\}} c_2\},$
the latest transition not in $\mathrm{tr}(\mathcal{B})$ is $b_1 \xrightarrow{\{a_0\}} b_0$ at step 4; $\mathrm{cb}(\pi, 4, b) = (2, 4)$, and as there is no transitions involving b between steps 3 and 4, $\Psi = \{(b, 2, 4)\}$; therefore, the sequence
$\varpi = \{a_0 \xrightarrow{\{b_0\}} a_1\}, \{c_0 \xrightarrow{\{a_1\}} c_1\}, \{a_1 \xrightarrow{\emptyset} a_0\}, \{\}, \{c_1 \xrightarrow{\{b_0\}} c_2\}$
is valid sub-trace of π reacing c_2, hence π is not minimal.

In conclusion, if π is a minimal trace for $g_\top$ reachability from state s, then, $\mathrm{tr}(\pi) \subset \mathrm{tr}(\mathcal{B})$. □

Model		# tr	# states	NuSMV time	NuSMV mem	ITS time	ITS mem
Egf-r (20)	normal	68	4.200	0.1s	15Mb	0.35s	19Mb
	reduced	**43**	**722**	**0.03s**	**11Mb**	**0.13s**	**8Mb**
EGF-r (104) profile 1	normal	378	$\approx 10^7$	75s	2.1Gb	0.8s	750Mb
	reduced	**0**	**1**				
EGF-r (104) profile 2	normal	378	$> 8.10^{14}$	KO	KO	540s	1.5Gb
	reduced	**211**	$\approx 6.10^7$	**52s**	**100Mb**	**3.4s**	**100Mb**
TCell-r (94)	normal	217	?	KO	KO	KO	KO
	reduced	**42**	**60.000**	**10s**	**190Mb**	**0.25s**	**15Mb**

Table 1: Experiments for goal-oriented reduction as pre-precessing for model-checking reachability properties. Each model is identified by the system, the number of automata (within parentheses), and a profile specifying the initial state and the reachability goal. "#tr" is the number of transitions in the automata network specification; "#states" the number of reachable states from the specified initial state; "KO" indicates an execution running out of time (30 minutes). Computation times where obtained on an Intel® Core™ i7 3.4GHz CPU.

4 Experiments

In this section, we show potential practical applications of the goal-oriented reduction introduced by this paper.

We conducted experiments on automata networks encoding Boolean networks that model dynamics of biological networks extracted from the literature in systems biology. For different initial states, and for different reachability goal, we compared the size of the reachable state space, and the time and memory consumption of classical model-checking tools.

The selected networks are models of signalling pathways, namely of the *Epidermal Growth Factor receptors* (EGF-r) with two different models, one comprising 20 interacting components [20] and another, more detailed, gathering 104 automata [21]; and of the *T-Cell receptors* (TCell-r) with a network between 94 automata. The automata networks result from automatic translations from Boolean network specification in the mentioned references. All the automata have 2 local states (modelling the inactivity and activity of related component) and local transitions within each automaton typically depends on the state of a few others automata, generally between 1 and 5 other automata. The mentioned models contain feedback circuits that make their transient dynamics non-trivial. In those models, we selected different initial states, acting for different state of the receptors, and selected a component (typically at the bottom of the signalling pathways) for checking the reachability of its activation.

The model-checking for the reachability properties is performed with two softwares relying on symbolic computations of the reachable state space: NuSMV [5] which combines BDD and SAT approaches for synchronous systems; and ITS (using libddd) [9, 12] which implements efficient decision diagram data structures and efficient handling of formalisms like Petri nets. It is worth noting that NuSMV implements the *cone of influence* reduction [4] which removes variables not involved in the property. In both cases, the transition systems specified as input of these tools is an encoding of the asynchronous semantics of the automata networks, where steps (definition 2) are always composed of only one transition.

Table 1 summarizes the results on the reachable state space, time and memory consumptions of the model checkers on the selected models and properties, before and after goal-oriented reduction[1]. The reduction is performed using a prototype implementation provided as part of the Pint software [16]. Because automata have very few internal states, the complexity of the reduction is low, computations took around a few tenth of a seconds.

Experiments show a remarkable gain in tractability for the model checking of models after reduction due to the removal of numerous transitions from the automata. In the case of the model EGF-r (104), the reduction removed all the transitions for first profile, resulting in an empty model. Such a behaviour can occur when the local causality analysis statically concludes that the reachability goal is impossible, i.e.,

[1]Scripts and models available at `http://loicpauleve.name/gored-experiments.tbz2`

the necessary condition of section 3.1 is not satisfied. In the case of TCell-r model, the dynamics before reduction is too broad for a raw exhaustive analysis, whereas the dynamics after reduction makes the model checking almost instantaneous with ITS.

5 Discussion

This paper introduces a new reduction for automata networks parametrized by a reachability property of the form: from a state s there exists a trace which lead to a state where a given automaton g is in state $g_\top$. The reduction is achieved by a static analysis of the local causality in the automata network to remove transitions within individual automata. Our goal-oriented reduction takes advantage of the component-based (automata) specification of the model and of the explicit transitions between their local state to efficiently decompose the reachability properties locally to each automaton. The overall complexity is polynomial in the total number of local states and transitions and exponential with the number of local states within one automaton. Therefore, the procedure can be extremely efficient when applied on automata networks that gathers numerous automata, but where each automaton has a few local states.

We demonstrated that the reduction preserves all the minimal traces satisfying the reachability property under a general concurrent semantics which allows at each step simultaneous transitions of an arbitrary number of automata. Hence, the minimal trace preservation is ensured for any stricter semantics, ranging from the fully asynchronous to maximally parallel transitions. The reachability properties considered here focus on the reachability of a local state of a single automaton. One can remark that reachability properties accounting for sequential reachability properties between (sub)states of the network can be straightforwardly encoded using an extra automaton where a local state $\top$ is reached if and only if the former reachability property is satisfied.

Applied to models of biological networks, the reduction reveals a significant shrinkage of the reachable state space, enhancing the scalability of analysis like model checking. In the performed experiments, the reduction has been used as a simple pre-processing step for the model checking. Part of the efficiency of the reduction depends on the ability to detect impossible intermediate reachability properties. For that purpose, our current implementation uses a simple static analysis, based on prior work, and further work may embed the checking for stronger necessary conditions, which may further increase the effectiveness of the goal-oriented reduction. One could also think about performing the reduction on the fly, during the state space exploration, expecting a stronger pruning. Although the complexity of the reduction is low, such an on-the-fly reduction would benefit from heuristics to indicate when a new reduction step may be worth to apply.

Finally, as the reduction procedure removes local transitions, it can be easily combined with other reduction techniques. In the scope of applications to systems biology, impact of the reduction on the topology of the network and its combination with Boolean and multi-valued network reductions is yet to be explored.

References

[1] Julio Aracena, Eric Goles, Andrés Moreira, and Lilian Salinas. On the robustness of update schedules in boolean networks. *Biosystems*, 97(1):1 – 8, 2009.

[2] Luca Bernardinello and Fiorella De Cindio. A survey of basic net models and modular net classes. In Grzegorz Rozenberg, editor, *Advances in Petri Nets 1992*, volume 609 of *Lecture Notes in Computer Science*, pages 304–351. Springer Berlin / Heidelberg, 1992.

[3] G. Berthelot and Lri-Iie. Checking properties of nets using transformations. In G. Rozenberg, editor, *Advances in Petri Nets 1985*, volume 222 of *Lecture Notes in Computer Science*, pages 19–40. Springer Berlin Heidelberg, 1986.

[4] Armin Biere, Edmund Clarke, Richard Raimi, and Yunshan Zhu. Verifying safety properties of a powerpc microprocessor using symbolic model checking without bdds. In *In Proc. 11 th Int. Conf. on Computer Aided Verification*, pages 60–71. Springer-Verlag, 1999.

[5] Alessandro Cimatti, Edmund Clarke, Enrico Giunchiglia, Fausto Giunchiglia, Marco Pistore, Marco Roveri, Roberto Sebastiani, and Armando Tacchella. NuSMV 2: An opensource tool for symbolic model checking. In *Computer Aided Verification*, volume 2404 of *Lecture Notes in Computer Science*, pages 241–268. Springer Berlin / Heidelberg, 2002.

[6] Vincent Danos, Jérôme Feret, Walter Fontana, Russell Harmer, Jonathan Hayman, Jean Krivine, Christopher D. Thompson-Walsh, and Glynn Winskel. Graphs, rewriting and pathway reconstruction for rule-based models. In Deepak D'Souza, Telikepalli Kavitha, and Jaikumar Radhakrishnan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012*, volume 18 of *LIPIcs*, pages 276–288. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012.

[7] Jerome Feret, Heinz Koeppl, and Tatjana Petrov. Stochastic fragments: A framework for the exact reduction of the stochastic semantics of rule-based models. *International Journal of Software and Informatics*, 7(4):527 – 604, 2013.

[8] Serge Haddad and Jean-François Pradat-Peyre. New efficient Petri nets reductions for parallel programs verification. *Parallel Processing Letters*, 16(1):101–116, March 2006.

[9] Alexandre Hamez, Yann Thierry-Mieg, and Fabrice Kordon. Building efficient model checkers using hierarchical set decision diagrams and automatic saturation. *Fundam. Inf.*, 94(3-4):413–437, 2009.

[10] Ryszard Janicki, Peter E. Lauer, Maciej Koutny, and Raymond Devillers. Concurrent and maximally concurrent evolution of nonsequential systems. *Theoretical Computer Science*, 43(0):213 – 238, 1986.

[11] Robert P Kurshan. *Computer-aided verification of coordinating processes: the automata-theoretic approach*. Princeton university press, 1994.

[12] LIP6/Move. the libDDD environment, libDDD. `http://ddd.lip6.fr`.

[13] Claire Loiseaux, Susanne Graf, Joseph Sifakis, Ahmed Bouajjani, Saddek Bensalem, and David Probst. Property preserving abstractions for the verification of concurrent systems. *Formal methods in system design*, 6(1):11–44, 1995.

[14] Guillaume Madelaine, Cédric Lhoussaine, and Joachim Niehren. Attractor Equivalence: An Observational Semantics for Reaction Networks. In *First International Conference on Formal Methods in Macro-Biology*, Lecture Notes in Bioinformatics, Nouméa, New Caledonia, September 2014. Springer-Verlag.

[15] Aurélien Naldi, Elisabeth Remy, Denis Thieffry, and Claudine Chaouiya. Dynamically consistent reduction of logical regulatory graphs. *Theoretical Computer Science*, 412(21):2207 – 2218, 2011.

[16] Loïc Paulevé. PINT - Static analyzer for dynamics of automata networks, `http://loicpauleve.name/pint`.

[17] Loïc Paulevé, Geoffroy Andrieux, and Heinz Koeppl. Under-approximating cut sets for reachability in large scale automata networks. In Natasha Sharygina and Helmut Veith, editors, *Computer Aided Verification*, volume 8044 of *Lecture Notes in Computer Science*, pages 69–84. Springer Berlin Heidelberg, 2013.

[18] Loïc Paulevé, Morgan Magnin, and Olivier Roux. Static analysis of biological regulatory networks dynamics using abstract interpretation. *Mathematical Structures in Computer Science*, 22(04):651–685, 2012.

[19] Lutz Priese and Harro Wimmel. A uniform approach to true-concurrency and interleaving semantics for petri nets. *Theoretical Computer Science*, 206(1–2):219 – 256, 1998.

[20] Ozgur Sahin, Holger Frohlich, Christian Lobke, Ulrike Korf, Sara Burmester, Meher Majety, Jens Mattern, Ingo Schupp, Claudine Chaouiya, Denis Thieffry, Annemarie Poustka, Stefan Wiemann, Tim Beissbarth, and Dorit Arlt. Modeling ERBB receptor-regulated G1/S transition to find novel targets for de novo trastuzumab resistance. *BMC Systems Biology*, 3(1), 2009.

[21] Regina Samaga, Julio Saez-Rodriguez, Leonidas G. Alexopoulos, Peter K. Sorger, and Steffen Klamt. The logic of egfr/erbb signaling: Theoretical properties and analysis of high-throughput data. *PLoS Comput Biol*, 5(8):e1000438, 08 2009.

[22] Philippe Schnoebelen and Natalia Sidorova. Bisimulation and the reduction of petri nets. In *Application and Theory of Petri Nets 2000*, volume 1825 of *Lecture Notes in Computer Science*, pages 409–423. Springer Berlin Heidelberg, 2000.

[23] Joseph Sifakis. Property preserving homomorphisms of transition systems. In Edmund Clarke and Dexter Kozen, editors, *Logics of Programs*, volume 164 of *Lecture Notes in Computer Science*, pages 458–473. Springer Berlin Heidelberg, 1984.