Under-approximating Cut Sets for Reachability in Large Scale Automata Networks

Loïc Paulevé, Geoffroy Andrieux, Heinz Koeppl

Under-approximating Necessary Processes for Reachability in Discrete Dynamical Systems

Loïc Paulevé[1], Geoffroy Andrieux[2], Heinz Koeppl[1]

[1] ETH Zürich, Switzerland.
[2] IRISA Rennes, France.

Abstract. In the scope of discrete finite-state models of interacting components, and calling the local state of a component a process, we present an algorithm for extracting sets of processes that are necessary for achieving the reachability of a given process. Those sets are referred to as cut sets and are computed from a particular abstract causality structure, so-called Graph of Local Causality, inspired from previous work and generalised here to Communicating Finite State Machines. We apply this new method to a very large model of biological interactions, providing sets of biological entities that, if de-activated, prevent molecules of interest to be active, up to the correctness of the model.

1 Introduction

Aiming at understanding and, ultimately, controlling physical systems, one generally constructs dynamical models of the known interactions between the components of the system. Because parts of those physical processes are still unknown or ignored, dynamics of such models have to be interpreted as an over-approximation of the real system dynamics: any (observed) behaviour of the real system has to have a matching behaviour in the abstract model, the converse being potentially false. In such a setting, a valuable contribution of formal methods on abstract models of physical systems resides in the ability to prove the impossibility of particular behaviours.

Given a discrete finite-state model of interacting components, we are interested in the control of reachability of the local state, referred to as process, of a given component from a partially-determined initial global state.

In this paper, we present an algorithm to extract sets of processes that are necessary to achieve the wanted reachability: informally, each trace leading to the reachability of interest has to involve, at one point, at least one process of such a set. Those sets are referred to as cut sets, and we limit ourselves to N-sets, *i.e.* having a maximum cardinality of N.

Applied to a model of a real system where the reachability of a process of interest has been observed, disabling all the processes referenced in a cut set should make the process reachability impossible from delimited initial states. The contrary implies that the abstract model is not an over-approximation of the real concrete system.

The computation of the cut N-sets takes advantage of an abstraction of the formal model which highlights some of the causalities of processes reachability. This results in a causality structure called a *Graph of Local Causality*, which is inspired by [1], and that we generalise here to Communicating Finite State Machines [2] (CFSMs).

The proposed algorithm aims at being tractable on systems composed of a very large number of interacting components, but each of them having a small number of processes (local states). Our method principally overcomes two challenges: (1) prevent a complete enumeration of candidate N-sets, which is intractable when having a large number of components; (2) prevent the use of model-checking techniques to check if disabling a set of processes break the reachability of the given process. It inherently handles partially-determined initial states: the resulting cut N-set of processes are proven to be necessary for the reachability of the process of interest from *any* of the supplied admissible initial states.

In order to demonstrate the applicability of our approach, we perform the search for cut N-sets of processes within a very large model of biological processes (relating more than 9000 components). Despite the very high combinatoric of possible dynamics of such a system, a prototype implementation of our algorithm manages to compute up to the 5-sets of necessary processes within a few minutes.

Related work and limitations. The aim of the presented method is somehow similar to the generation of minimal cut sets in fault trees [3,4], as the structure representing reachability causality contains both *and* and *or* connectors. However, the major difference is that we are here dealing with cyclic directed graphs which prevents the above mentioned methods to be straightforwardly applied. [5] develops a method for generating minimal cut sets (also called intervention sets) dedicated to biochemical reactions networks, hence involving cycling dependencies. This method has been later generalised to boolean models of signalling networks [6]. Those algorithms are mainly based on the enumeration of possible candidates, with techniques to reduce the search state space, for example by exploiting symmetry of dynamics. Our method follows a different approach than [5,6] in particular by not relying on candidate enumeration but on minimal cut sets propagations and combinations driven by our special structure of causality. In addition our method is generic to any dynamical discrete system, but relies on dynamics over-approximation which leads to under-approximating the minimal cut sets for reachability. Finally, we focus on finding the minimal cut sets for the reachability of only *one* process (i.e. state of one component). Nevertheless when our algorithm terminates the minimal cut sets for the (independent) reachability of all processes referenced in the causality structure are computed.

Outline. Sect. 2 introduces a generic characterisation of a *Graph of Local Causality* with respect to CFSMs; Sect. 3 states and sketches the proof of the algorithm for extracting a subset of minimal N-sets of processes necessary for the reachability of a given process. Sect. 4 shows a real-case application of this new method to

the analysis of a very large model of a biological system. Finally, Sect. 5 discusses the results presented and some of their possible extensions.

Notations. $\wedge$ and $\vee$ are the usual logical *and* and *or* connectors. $[1;n] = \{1, \cdots, n\}$. Given a finite set A, $\#A$ is the cardinality of A; $\wp(A)$ is the power set of A; $\wp^{\leq N}(A)$ is the set of all subsets of A with cardinality at most N. Given sets $A^1, \ldots, A^n$, $\bigcup_{i\in[1;n]} A^i$ is the union of those sets, with the empty union $\bigcup_\emptyset \triangleq \emptyset$; and $A^1 \times \cdots \times A^n$ is the usual Cartesian product. Given sets of sets $B^1, \ldots, B^n \in \wp(\wp(A))$, $\widetilde{\prod}_{i\in[1;n]} B^i \triangleq B^1 \tilde{\times} \cdots \tilde{\times} B^n \in \wp(\wp(A))$ is the *sets of sets product* where $\{e_1, \ldots, e_n\} \tilde{\times} \{e'_1, \ldots, e'_m\} \triangleq \{e_i \cup e'_j \mid i \in [1;n] \wedge j \in [1;m]\}$. In particular $\forall (i,j) \in [1;n] \times [1;m]$, $B^i \tilde{\times} B^j = B^j \tilde{\times} B^i$ and $\emptyset \tilde{\times} B^i = \emptyset$. The empty sets of sets product $\widetilde{\prod}_\emptyset \triangleq \{\emptyset\}$. If $M : A \mapsto B$ is a mapping from elements in A to elements in B, $M(a)$ is the value in B mapped to $a \in A$; $M\{a \mapsto b\}$ is the mapping M where $a \in A$ now maps to $b \in B$.

2 Graph of Local Causality

We first give basic definitions of process disabling, context and process reachability upon Communicating Finite State Machines; then we define the local causality of an objective (local reachability), and the *Graph of Local Causality*. A simple example is given at the end of the section.

2.1 Finite Discrete Dynamical Systems

We consider any dynamical system $\mathcal{S}ys$ relating a finite number of interacting components, each of these component having a finite number of local states. Without loss of generality, we build our definitions upon *Communicating Finite State Machines* (CFSMs, Def. 1). CFSMs $(\Sigma, S, \mathcal{L}, T)$ relate a finite number of interacting finite state automata, where the global state of the system is the gathering of the local state of composing automata. Each automaton can evolve on its own following internal transitions labelled with ϵ, or by changing of state synchronously with at least one other automaton by applying transitions with a synchronisation label $\ell \in \mathcal{L}$.

Definition 1 (CFSMs $(\Sigma, S, \mathcal{L}, T)$). *A set of Communicating Finite State Machines (CFSMs) is defined by a tuple $(\Sigma, S, \mathcal{L}, T)$ where*

- $\Sigma = \{a, b, \ldots, z\}$ *is the finite set of automata identifiers;*
- $S = \prod_{a\in\Sigma}[1;k_a]$ *the finite set of global states; k_a being the number of local states of automaton $a \in \Sigma$. We note $S(a) \triangleq [1;k_a]$; and given $s \in S$ we note $s(a)$ the local state of automaton a.*
- $\mathcal{L} = \{\ell_1, \ldots, \ell_m\}$ *is the finite set of synchronisation labels;*
- $T = \{a \mapsto T_a \mid a \in \Sigma\}$*, where $\forall a \in \Sigma, T_a \subset [1;k_a] \times (\mathcal{L} \cup \{\epsilon\}) \times [1;k_a]$, is the mapping from automata to their finite set of local transitions.*
  We note $i \xrightarrow{\ell} j \in T(a) \overset{\Delta}{\Leftrightarrow} (i,\ell,j) \in T_a$ and $a_i \xrightarrow{\ell} a_j \in T \overset{\Delta}{\Leftrightarrow} i \xrightarrow{\ell} j \in T(a)$.

The set of processes is defined as $\mathbf{Proc} \triangleq \{a_i \mid a \in \Sigma \wedge i \in [1; k_a]\}$.
The global transition relation $\rightarrow \subset S \times S$ *is defined as:*

$$s \rightarrow s' \overset{\Delta}{\Leftrightarrow} \begin{cases} \exists a \in \Sigma : s(a) \xrightarrow{\epsilon} s'(a) \in T(a) \wedge \forall b \in \Sigma, b \neq a, s'(b) = s(b) \quad or, \\ \exists \ell \in \mathcal{L}, \forall a \in \Sigma, s'(a) \neq s(a) \Leftrightarrow s(a) \xrightarrow{\ell} s'(a) \in T(a) \\ \quad \wedge \exists a, b \in \Sigma, a \neq b : s'(a) \neq s(a) \wedge s'(b) \neq s(b). \end{cases}$$

We refer to the local state of an automaton (component) as a *process*: $a_i \in \mathbf{Proc}$ is the process representing the local state $i \in S(a)$ of automaton $a \in \Sigma$.

Given a dynamical system $\mathcal{S}ys$ and a set of its processes $ps \subseteq \mathbf{Proc}$, $\mathcal{S}ys \ominus ps$ refers to the system where all processes in ps have been disabled, i.e. they can not be involved in any transition. In the case of CFSMs, this is equivalent to removing all transitions outgoing from processes (states) referenced in ps (Def. 2).

Definition 2 (Process Disabling). *Given* $\mathcal{S}ys = (\Sigma, S, \mathcal{L}, T)$ *and* $ps \in \wp(\mathbf{Proc})$, $\mathcal{S}ys \ominus ps \triangleq (\Sigma, S, \mathcal{L}, T')$ *where* $T' = \{a_i \xrightarrow{\ell} a_j \in T \mid a_i \notin ps\}$.

From a set of acceptable initial states delimited by a *context* ς (Def. 3), we say a given process $a_j \in \mathbf{Proc}$ is reachable if and only if there exists a finite number of transitions in $\mathcal{S}ys$ leading to a global state where a_j is present (Def. 4).

Definition 3 (Context ς). *Given CFSMs* $(\Sigma, S, \mathcal{L}, T)$*, a context* ς *is a mapping from each automaton* $a \in \Sigma$ *to a non-empty subset of its local states:* $\forall a \in \Sigma, \varsigma(a) \in \wp(S(a)) \wedge \varsigma(a) \neq \emptyset$.

Definition 4 (Process reachability). *Given CFSMs* $(\Sigma, S, \mathcal{L}, T)$ *and a context* ς*, the process* $a_j \in \mathbf{Proc}$ *is* reachable *from* ς *if and only if* $\exists s_0, \ldots, s_m \in S$ *such that* $\forall a \in \Sigma, s_0(a) \in \varsigma(a)$*, and* $s_0 \rightarrow \cdots \rightarrow s_m$*, and* $s_m(a) = j$.

2.2 Local Causality

Locally reasoning within one automaton, the reachability of process a_j from ς can be expressed as the reachability of a_j from a process $a_i \in \varsigma(a)$. This local reachability specification is referred to as an *objective* noted $a_i \rightarrow^* a_j$ (Def. 5)

Definition 5 (Objective). *Given CFSMs* $(\Sigma, S, \mathcal{L}, T)$*, the reachability of process* a_j *from* a_i *is called an* objective *and is denoted* $a_i \rightarrow^* a_j$*. The set of all objectives is referred to as* $\mathbf{Obj} \triangleq \{a_i \rightarrow^* a_j \mid (a_i, a_j) \in \mathbf{Proc} \times \mathbf{Proc}\}$.

Given an objective $P = a_i \rightarrow^* a_j \in \mathbf{Obj}$, we define $\mathsf{sol}(P)$ the *local causality* of P (Def. 6): each element $ps \in \mathsf{sol}(P)$ is a subset of processes, referred to as a (local) solution for P, which are all involved at some times prior to the reachability of a_j. This set of solutions is sound if when at least one process in *each* solution is disabled, the reachability of a_j from any global state containing a_i is impossible (Property 1). Note that if $\mathsf{sol}(P) = \{\{a_i\} \cup ps^1, \ldots, ps^m\}$ is sound, $\mathsf{sol}'(P) = \{ps^1, \ldots, ps^m\}$ is also sound. $\mathsf{sol}(a_i \rightarrow^* a_j) = \emptyset$ implies that a_j can never be reached from a_i, and $\forall a_i \in \mathbf{Proc}, \mathsf{sol}(a_i \rightarrow^* a_i) \triangleq \{\emptyset\}$.

Definition 6. $\mathsf{sol} : \mathbf{Obj} \mapsto \wp(\wp(\mathbf{Proc}))$ *is a mapping from objectives to set of set of processes such that* $\forall P \in \mathbf{Obj}, \forall ps \in \mathsf{sol}(P), \nexists ps' \in \mathsf{sol}(P), ps \neq ps'$ *such that* $ps' \subset ps$*. The set of these mappings is noted* $\mathbf{Sol} \triangleq \{\langle P, ps\rangle \mid ps \in \mathsf{sol}(P)\}$.

Property 1 (sol *soundness).* $\mathsf{sol}(a_i \rightarrow^* a_j) = \{ps^1, \ldots, ps^n\}$ is a sound set of solutions if and only if $\forall kps \in \widetilde{\prod}_{i\in[1;n]} ps^i$, a_j is not reachable from any state $s \in S$ such that $s(a) = i$ in $\mathcal{S}ys \ominus kps$.

In the rest of this paper we assume that Property 1 is verified, and consider sol computation out of the scope of this paper.

Nevertheless, we briefly describe a construction of a sound $\mathsf{sol}(a_i \rightarrow^* a_j)$ from CFSMs $(\Sigma, S, \mathcal{L}, T)$; an example is given at the end of this section. For each acyclic sequence $a_i \xrightarrow{\ell_1} \ldots \xrightarrow{\ell_m} a_j$ of local transitions in $T(a)$, and by defining $\mathsf{ext}_a(\ell) \triangleq \{b_j \in \mathbf{Proc} \mid b_j \xrightarrow{\ell} b_k \in T, b \neq a\}$, we set $ps \in \widetilde{\prod}_{\ell\in\{\ell_1,\ldots,\ell_m\}\setminus\{\epsilon\}} \mathsf{ext}_a(\ell) \Rightarrow ps \in \mathsf{sol}(a_i \rightarrow^* a_j)$, up to sursets removing. One can easily show that Property 1 is verified with such a construction. The complexity of this construction is exponential in the number of local states of automata and polynomial in the number of automata. Alternative constructions may also provide sound (and not necessarily equal) sol.

2.3 Graph of Local Causality

Given a process $a_j \in \mathbf{Proc}$ and an initial context ς, the reachability of a_i is equivalent to the realization of any objective $a_i \rightarrow^* a_j$, with $a_i \in \varsigma(a)$. By definition, if a_j is reachable from ς, there exists $ps \in \mathsf{sol}(a_i \rightarrow^* a_j)$ such that, $\forall b_k \in ps$, b_k is reachable from ς.

The (directed) *Graph of Local Causality* (GLC, Def. 7) relates this recursive reasoning from a given set of processes $\omega \subseteq \mathbf{Proc}$ by linking every process a_j to all objectives $a_i \rightarrow^* a_j, a_i \in \varsigma(a)$; every objective P to its solutions $\langle P, ps\rangle \in \mathbf{Sol}$; every solution $\langle P, ps\rangle$ to its processes $b_k \in ps$. A GLC is said to be valid if sol is sound for all referenced objectives (Property 2).

Definition 7 (Graph of Local Causality). *Given a context* ς *and a set of processes* $\omega \subseteq \mathbf{Proc}$*, the* Graph of Local Causality *(GLC)* $\mathcal{A}^\omega_\varsigma \triangleq (V^\omega_\varsigma, E^\omega_\varsigma)$*, with* $V^\omega_\varsigma \subseteq \mathbf{Proc} \cup \mathbf{Obj} \cup \mathbf{Sol}$ *and* $E^\omega_\varsigma \subseteq V^\omega_\varsigma \times V^\omega_\varsigma$*, is the smallest structure satisfying:*

$$
\begin{aligned}
\omega &\subseteq V^\omega_\varsigma \\
a_i \in V^\omega_\varsigma \cap \mathbf{Proc} &\Leftrightarrow \{(a_i, a_j \rightarrow^* a_i) \mid a_j \in \varsigma\} \subseteq E^\omega_\varsigma \\
a_i \rightarrow^* a_j \in V^\omega_\varsigma \cap \mathbf{Obj} &\Leftrightarrow \{(a_i \rightarrow^* a_j, \langle a_i \rightarrow^* a_j, ps\rangle) \mid \langle a_i \rightarrow^* a_j, ps\rangle \in \mathbf{Sol}\} \subseteq E^\omega_\varsigma \\
\langle P, ps\rangle \in V^\omega_\varsigma \cap \mathbf{Sol} &\Leftrightarrow \{(\langle P, ps\rangle, a_i) \mid a_i \in ps\} \subseteq E^\omega_\varsigma \quad .
\end{aligned}
$$

Property 2 (Valid Graph of Local Causality). A GLC $\mathcal{A}^\omega_\varsigma$ is *valid* if, $\forall P \in V^\omega_\varsigma \cap \mathbf{Obj}$, $\mathsf{sol}(P)$ is sound.

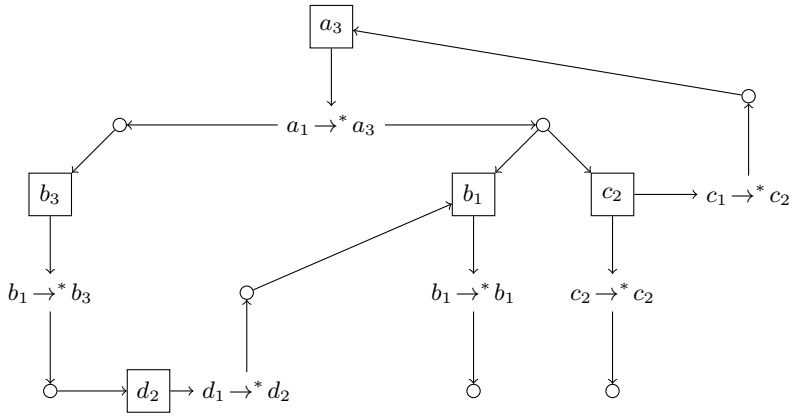

Fig. 1. Example of Graph of Local Causality that is valid for the CFSMs defined in Example 1

This structure can be constructed from processes in ω and by iteratively adding the imposed children. It is worth noticing that this graph can contain cycles. In the worst case, $\#V_\varsigma^\omega = \#\mathbf{Proc} + \#\mathbf{Obj} + \#\mathbf{Sol}$ and $\#E_\varsigma^\omega = \#\mathbf{Obj} + \#\mathbf{Sol} + \sum_{\langle P, ps\rangle \in \mathbf{Sol}} \#ps$.

Example 1. Fig. 1 shows an example of GLC. Processes are represented by boxed nodes and elements of **Sol** by small circles.

For instance, such a GLC is valid for the following CFSMs $(\Sigma, S, \mathcal{L}, T)$, with initial context $\varsigma = \{a \mapsto \{1\}; b \mapsto \{1\}; c \mapsto \{1,2\}; d \mapsto \{2\}\}$:

$$\Sigma = \{a, b, c, d\} \qquad \mathcal{L} = \{\ell_1, \ell_2, \ell_3, \ell_4, \ell_5, \ell_6\}$$
$$S(a) = [1;3] \qquad T(a) = \{1 \xrightarrow{\ell_2} 2; 2 \xrightarrow{\ell_3} 3; 1 \xrightarrow{\ell_1} 3; 3 \xrightarrow{\ell_4} 2\}$$
$$S(b) = [1;3] \qquad T(b) = \{1 \xrightarrow{\ell_2} 2; 1 \xrightarrow{\ell_5} 3; 1 \xrightarrow{\ell_6} 1; 3 \xrightarrow{\ell_1} 2\}$$
$$S(c) = [1;2] \qquad T(c) = \{1 \xrightarrow{\ell_4} 2; 2 \xrightarrow{\ell_3} 1\}$$
$$S(d) = [1;2] \qquad T(d) = \{1 \xrightarrow{\ell_6} 2; 2 \xrightarrow{\ell_5} 1\}$$

For example, within automata a, there are two acyclic sequences from 1 to 3: $1 \xrightarrow{\ell_2} 2 \xrightarrow{\ell_3} 3$ and $1 \xrightarrow{\ell_1} 3$. Hence, if a_3 is reached from a_1, then necessarily, one of these two sequences has to be used (but not necessarily consecutively). For each of these transitions, the synchronisation label is shared by exactly one process in another automaton: b_1, c_2, b_3 for ℓ_2, ℓ_3, ℓ_1, respectively. Therefore, if a_3 is reached from a_1, then necessarily either both b_1 and c_2, or b_3 have been reached before. Hence $\mathsf{sol}(a_1 \to^* a_3) = \{\{b_1, c_2\}, \{b_3\}\}$ is sound, as disabling either b_1 and b_3, or c_2 and b_3, would remove any possibility to reach a_3 from a_1.

3 Necessary Processes for Reachability

We assume a global GLC $\mathcal{A}^\omega_\varsigma = (V^\omega_\varsigma, E^\omega_\varsigma)$, with the usual accessors for the direct relations of nodes:

$$\mathsf{children} : V^\omega_\varsigma \mapsto \wp(V^\omega_\varsigma) \qquad \mathsf{parents} : V^\omega_\varsigma \mapsto \wp(V^\omega_\varsigma)$$
$$\mathsf{children}(n) \triangleq \{m \in V^\omega_\varsigma \mid (n,m) \in E^\omega_\varsigma\} \quad \mathsf{parents}(n) \triangleq \{m \in V^\omega_\varsigma \mid (m,n) \in E^\omega_\varsigma\}$$

Given a set of processes $\mathcal{O}bs \subseteq \mathbf{Proc}$, this section introduces an algorithm computing upon $\mathcal{A}^\omega_\varsigma$ the set $\mathbb{V}(a_i)$ of minimal cut N-sets of processes in $\mathcal{O}bs$ that are necessary for the independent reachability of each process $a_i \in \mathbf{Proc} \cap V^\omega_\varsigma$. The minimality criterion actually states that $\forall ps \in \mathbb{V}(a_i)$, there is no different $ps' \in \mathbb{V}(a_i)$ such that $ps' \subset ps$.

Assuming a first valuation $\mathbb{V}$ (Def. 8) associating to each node a set of (minimal) cut N-sets, the set of cut N-sets for the node n can be refined following $\mathsf{update}(\mathbb{V}, n)$ (Def. 9):

- if n is a solution $\langle P, ps\rangle \in \mathbf{Sol}$, it is sufficient to prevent the reachability of *any* process in ps to cut n; therefore, the cut N-sets results from the union of the cut N-sets of n children (all processes).
- If n is an objective $P \in \mathbf{Obj}$, all its solutions (in $\mathsf{sol}(P)$) have to be cut in order to ensure that P is not realizable: hence, the cut N-sets result from the product of children cut N-sets (all solutions).
- If n is a process a_i, it is sufficient to cut all its children (all objectives) to prevent the reachability of a_i from any state in the context ς. In addition, if $a_i \in \mathcal{O}bs$, $\{a_i\}$ is added to the set of its cut N-sets.

Definition 8 (Valuation $\mathbb{V}$). *A* valuation $\mathbb{V} : V^\omega_\varsigma \mapsto \wp(\wp^{\leq N}(\mathcal{O}bs))$ *is a mapping from each node of* $\mathcal{A}^\omega_\varsigma$ *to a set of* N*-sets of processes.* $\mathbf{Val}$ *is the set of all valuations.* $\mathbb{V}_0 \in \mathbf{Val}$ *refers to the valuation such that* $\forall n \in V^\omega_\varsigma, \mathbb{V}_0(n) = \emptyset$.

Definition 9 ($\mathsf{update} : \mathbf{Val} \times V^\omega_\varsigma \mapsto \mathbf{Val}$).

$$\mathsf{update}(\mathbb{V}, n) \triangleq \begin{cases} \mathbb{V}\{n \mapsto \zeta^N(\bigcup_{m \in \mathsf{children}(n)} \mathbb{V}(m))\} & \text{if } n \in \mathbf{Sol} \\ \mathbb{V}\{n \mapsto \zeta^N(\widetilde{\prod}_{m \in \mathsf{children}(n)} \mathbb{V}(m))\} & \text{if } n \in \mathbf{Obj} \\ \mathbb{V}\{n \mapsto \zeta^N(\widetilde{\prod}_{m \in \mathsf{children}(n)} \mathbb{V}(m))\} & \text{if } n \in \mathbf{Proc} \setminus \mathcal{O}bs \\ \mathbb{V}\{n \mapsto \zeta^N(\{\{a_i\}\} \cup \widetilde{\prod}_{m \in \mathsf{children}(n)} \mathbb{V}(m))\} & \text{if } n \in \mathbf{Proc} \cap \mathcal{O}bs \end{cases}$$

where $\zeta^N(\{e_1, \ldots, e_n\}) \triangleq \{e_i \mid i \in [1;n] \wedge \#e_i \leq N \wedge \nexists j \in [1;n], j \neq i, e_j \subset e_i\}$, e_i *being sets,* $\forall i \in [1;n]$.

Starting with $\mathbb{V}_0$, one can repeatedly apply update on each node of $\mathcal{A}^\omega_\varsigma$ to refine its valuation. Only nodes where one of their children value has been modified should be considered for updating.

Hence, the order of nodes updates should follow the topological order of the GLC, where children have a lower rank than their parents (hence children are

Algorithm 1 $\mathcal{A}^{\omega}_{\varsigma}$-MINIMAL-CUT-NSETS

```
1: M ← V_ς^ω
2: 𝕍 ← 𝕍_0
3: while M ≠ ∅ do
4:     n ← arg min_{m∈M}{rank(m)}
5:     M ← M \ {n}
6:     𝕍' ← update(𝕍, n)
7:     if 𝕍'(n) ≠ 𝕍(n) then
8:         M ← M ∪ parents(n)
9:     end if
10:    𝕍 ← 𝕍'
11: end while
12: return 𝕍
```

treated before their parents). If the graph is actually acyclic, then it is sufficient to update the value of each node only once. In the general case, *i.e.* in the presence of Strongly Connected Components (SCCs) — nodes belonging to the same SCC have the same rank —, the nodes within a SCC have to be iteratively updated until the convergence of their valuation.

Algorithm 1 formalizes this procedure where $\mathsf{rank}(n)$ refers to the topological rank of n, as it can be derived from Tarjan's strongly connected components algorithm [7], for example. The node $n \in V^{\omega}_{\varsigma}$ to be updated is selected as being the one having the least rank amongst the nodes to update (delimited by $\mathcal{M}$). In the case where several nodes with the same lowest rank are in $\mathcal{M}$, they can be either arbitrarily or randomly picked. Once picked, the value of n is updated. If the new valuation of n is different from the previous, the parents of n are added to the list of nodes to update (lines 6-8 in Algorithm 1).

Lemma 1 states the convergence of Algorithm 1 and Theorem 1 its correctness: for each process $a_i \in V^{\omega}_{\varsigma} \cap \mathbf{Proc}$, each set of processes $kps \in \mathbb{V}(a_i)$ (except $\{a_i\}$ singleton) references the processes that are all necessary to reach in order to reach a_i from any state in ς. Hence, if all processes in kps are disabled in the dynamical system $\mathcal{S}ys$, a_i is not reachable from any state in ς.

Lemma 1. $\mathcal{A}^{\omega}_{\varsigma}$-MINIMAL-CUT-NSETS *always terminates.*

Proof. Remarking that $\wp(\wp^{\leq N}(\mathcal{O}bs))$ is finite, defining a partial ordering such that $\forall v, v' \in \wp(\wp^{\leq N}(\mathcal{O}bs)), v \succeq v' \overset{\Delta}{\Leftrightarrow} \zeta^N(v) = \zeta^N(v \cup v')$, and noting $\mathbb{V}^k \in \mathbf{Val}$ the valuation after k iterations of the algorithm, it is sufficient to prove that $\mathbb{V}^{k+1}(n) \succeq \mathbb{V}^k(n)$. Let us define $v_1, v_2, v'_1, v'_2 \in \wp(\wp^{\leq N}(\mathcal{O}bs))$ such that $v_1 \succeq v'_1$ and $v_2 \succeq v'_2$. We can easily check that $v_1 \cup v_2 \succeq v'_1 \cup v'_2$ (hence proving the case when $n \in \mathbf{Sol}$). As $\zeta^N(v_1) = \zeta^N(v_1 \cup v'_1) \Leftrightarrow \forall e'_1 \in v'_1, \exists e_1 \in v_1 : e_1 \subseteq e'_1$, we obtain that $\forall (e'_1, e'_2) \in v'_1 \times v'_2, \exists (e_1, e_2) \in v_1 \times v_2 : e_1 \subseteq e'_1 \wedge e_2 \subseteq e'_2$. Hence $e_1 \cup e_2 \subseteq e'_1 \cup e'_2$, therefore $\zeta^N(v_1 \tilde{\times} v_2 \cup v'_1 \tilde{\times} v'_2) = \zeta^N(v_1 \tilde{\times} v_2)$, i.e. $v_1 \tilde{\times} v_2 \succeq v'_1 \tilde{\times} v'_2$; which proves the cases when $n \in \mathbf{Obj} \cup \mathbf{Proc}$.

Node	rank	$\mathbb{V}$
$\langle b_1 \rightarrow^* b_1, \emptyset \rangle$	1	$\emptyset$
$b_1 \rightarrow^* b_1$	2	$\emptyset$
b_1	3	$\{\{b_1\}\}$
$\langle d_1 \rightarrow^* d_2, \{b_1\} \rangle$	4	$\{\{b_1\}\}$
$d_1 \rightarrow^* d_2$	5	$\{\{b_1\}\}$
d_2	6	$\{\{b_1\}, \{d_2\}\}$
$\langle b_1 \rightarrow^* b_3, \{d_2\} \rangle$	7	$\{\{b_1\}, \{d_2\}\}$
$b_1 \rightarrow^* b_3$	8	$\{\{b_1\}, \{d_2\}\}$
b_3	9	$\{\{b_1\}, \{b_3\}, \{d_2\}\}$
$\langle a_1 \rightarrow^* a_3, \{b_3\} \rangle$	10	$\{\{b_1\}, \{b_3\}, \{d_2\}\}$
$\langle c_2 \rightarrow^* c_2, \emptyset \rangle$	11	$\emptyset$
$c_2 \rightarrow^* c_2$	12	$\emptyset$
c_2	13	$\{\{c_2\}\}$
$\langle a_1 \rightarrow^* a_3, \{b_1, c_2\} \rangle$	13	$\{\{b_1\}, \{c_2\}\}$
$a_1 \rightarrow^* a_3$	13	$\{\{b_1\}, \{b_3, c_2\}, \{c_2, d_2\}\}$
a_3	13	$\{\{a_3\}, \{b_1\}, \{b_3, c_2\}, \{c_2, d_2\}\}$
$\langle c_1 \rightarrow^* c_2, \{a_3\} \rangle$	13	$\{\{a_3\}, \{b_1\}, \{b_3, c_2\}, \{c_2, d_2\}\}$

Table 1. Result of the execution of Algorithm 1 on the GLC in Fig. 1

Theorem 1. *If $\mathcal{A}^{\omega}_{\varsigma}$ is a valid graph for causality abstraction, the valuation $\mathbb{V}$ returned by $\mathcal{A}^{\omega}_{\varsigma}$-*Minimal-Cut-NSets *verifies: $\forall a_i \in \mathbf{Proc} \cap V^{\omega}_{\varsigma}, \forall kps \in \mathbb{V}(a_i) \setminus \{\{a_i\}\}$, a_j is not reachable from ς within $\mathcal{S}ys \ominus kps$.*

Proof. By recurrence on the valuations $\mathbb{V}$: the above property is true at each iteration of the algorithm.

Example 2. Table 1 details the result of the execution of Algorithm 1 on the GLC defined in Fig. 1. Nodes receive a topological rank, identical ranks implying the belonging to the same SCC. The (arbitrary) scheduling of the updates of nodes within a SCC follows the order in the table. In this particular case, nodes are all visited once, as $\mathbb{V}(\langle c_2 \rightarrow^* c_2, \emptyset \rangle) \tilde{\times} \mathbb{V}(\langle c_1 \rightarrow^* c_2, \{a_3\} \rangle) = \emptyset$ (hence $\mathsf{update}(\mathbb{V}, c_2)$ does not change the valuation of c_2). Note that in general, several iterations of update may be required to reach a fix point.

4 Application to Systems Biology

In order to support the scalability of our approach, we apply the proposed algorithm to a very large model of biological interactions, actually extracted from the PID database [8] referencing various influences (complex formation, inductions (activations) and inhibitions, transcriptional regulation, etc.) between more than 9000 biological components (proteins, genes, ions, etc.).

Amongst the numerous biological components, the activation of some of them are known to control key mechanisms of the cell dynamics. Those activations are the consequence of intertwining signalling pathways and depend on the environment of the cell (represented by the presence of certain *entry-point* molecules).

Uncovering the environmental and intermediate components playing a major role in these signalling dynamics is of great biological interest.

A cut N-set for such biological models inform that at least one of the process in the cut N-set has to be present in order to achieve the wanted reachability. A process can represent, for instance, an active transcription factor or the absence of a certain protein. This provides potential therapeutic targets if the studied reachability is involved in a disease by preventing any process in a cut N-set to appear, for instance using gene knock-down or knock-up techniques.

The full PID database has been interpreted into the Process Hitting framework [9], a subclass of CFSMs from which the derivation of the GLC has been addressed in previous work [1]. The obtained model gathers components representing either biological entities modeled as boolean value (absent or present), or logical complexes. When a biological component has several competing regulators, we use of two different interpretations: all (resp. one of) the activators and none (resp. all but one of) the inhibitors have to be present in order to make the target component present. This leads to two different discrete models of PID that we refer as `whole_PID_AND` and `whole_PID_OR`, respectively.

Focusing on `whole_PID_OR`, the Process Hitting model relates 67746 processes split into 21513 components, the largest component grouping 4 processes. Such a system could actually generate 2^{33874} states. 3136 components act as environment specification, which in our boolean interpretation leads to 2^{3136} possible initial states, assuming all other components start in the absent state.

We focus on the (independent) reachability of active SNAIL transcription factor, involved in the epithelial to mesenchymal transition [10], and of active p15INK4b and p21CIP1 cyclin-dependent kinase inhibitors involved in cell cycle regulation [11]. The GLC relates 20045 nodes, including 5671 processes (biological or logical); it contains 6 SCCs with at least 2 nodes, the largest being composed of 10238 nodes and the others between 20 and 150.

Table 2 shows the results of a prototype implementation[3] of Algorithm 1 for the search of up to the 6-sets of biological processes. One can observe that the execution time grows very rapidly with N compared to the number of visited nodes. This can be explained by intermediate nodes having a large set of cut N-sets leading to a costly computation of products.

While the precise biological interpretation of found N-sets is out of the scope of this paper, we remark that the order of magnitude of the number of cut sets can be very different (more than 1000 cut 6-sets for SNAIL; none cut 6-sets for p21CIP1, except the gene that produces this protein). It supports a notion of robustness for the reachability of components, where the less cut sets, the more robust the reachability to various perturbations.

Applied to the `whole_PID_AND` model, our algorithm find in general much more cut N-sets, due to the conjunctive interpretation. This brings a significant increase in the execution time: the search up to the cut 5-sets took 1h, and the 6-sets leads to an out-of-memory failure.

[3] Implemented as part of the PINT software – `http://process.hitting.free.fr`. Models and command lines are provided in `http://loicpauleve.name/lata13.tbz2`

Model	N	Visited nodes	Exec. time	Nb. of resulting N-sets		
				$SNAIL_1$	$p15INK4b_1$	$p21CIP1_1$
whole_PID_OR	1	29022	0.9s	1	1	1
	2	36602	1.6s	+6	+6	+0
	3	44174	5.4s	+0	+92	+0
	4	54322	39s	+30	+60	+0
	5	68214	8.3m	+90	+80	+0
	6	90902	2.6h	+930	+208	+0

Table 2. Number of nodes visited and execution time of the search for cut N-sets of 3 processes. For each N, only the number of additional N-sets is displayed.

5 Discussion

We presented a new method to compute minimal cut sets for the reachability of processes within discrete dynamical systems from any state delimited by a provided so-called context. Those cut sets are sets of processes such that disabling all processes of a cut set within the system guarantees to prevent the reachability of the concerned process.

We consider finite systems with interacting components, each component having a finite number of local states, that we refer to as processes: one and only one process of each component is present at any time, giving the global state of the system. Upon such a system, we characterized a *Graph of Local Causality* (GLC) which relates the local reachability of one process from another in the same component to the (external) processes that are necessary for this reachability to occur. Note that several constructions of such a GLC are possible and depend on the semantics of the model. We gave an example of such a construction for Communicating Finite State Machines.

The proposed algorithm then propagates along the GLC the minimal cut sets following a topological order. This leads to an under-approximation of the cut N-sets for the reachability of any process referenced in the GLC from any state delimited in a supplied context. At the current time, our algorithm does not suggest any particular scheduling for the update of nodes within a strongly connected component of the GLC. Further work may propose heuristics on this scheduling in order to improve convergence time.

A prototype implementation of our algorithm has been successfully applied to the extraction of minimal cut sets within a very large model of biological system, involving more than 9000 interacting components. To our knowledge this is the first attempt of such a dynamical analysis for such large biological models. We note that most of the computation time is due to products between large sets of cut N-sets. To partially address this issue, we use of BDD-like structures to represent set of sets [12] on which we have specialized operations to stick to sets of N-sets. There is still room for improvement as our prototype does not implement any caching or variable re-ordering.

The work presented in this paper can be extended in several ways, notably with a *posterior enlarging of the cut sets*. Because the algorithm computes the cut N-sets for each node in the GLC, it is possible to construct *a posteriori* cut sets that are not limited to N processes by chaining them. For instance, let $kps \in \mathbb{V}(a_i)$ be a cut N-set for the process a_i, for each $b_j \in kps$ and $kps' \in \mathbb{V}(b_j)$, $(kps \setminus \{b_j\}) \cup kps'$ is a cut set for the process a_i. In our biological case study, this method could be recursively applied until cut sets are composed of only processes representing the environment input.

With respect to the defined computation of cut N-sets, one could also derive *static reductions* of the GLC. Indeed, some particular nodes and arcs of the GLC can be removed without affecting the final valuation of nodes. A simple example are nodes representing objectives having no solution: such nodes can be safely removed as they bring no candidate N-sets for parents processes. These reductions conduct to both speedup of the proposed algorithm but also to more reduced representations of reachability causality.

References

1. Paulevé, L., Magnin, M., Roux, O.: Static analysis of biological regulatory networks dynamics using abstract interpretation. Mathematical Structures in Computer Science **22**(04) (2012) 651–685
2. Brand, D., Zafiropulo, P.: On communicating finite-state machines. Journal of the ACM **30** (April 1983) 323–342
3. Lee, W.S., Grosh, D.L., Tillman, F.A., Lie, C.H.: Fault tree analysis, methods, and applications - a review. IEEE Transactions on Reliability **R-34** (1985) 194–203
4. Tang, Z., Dugan, J.: Minimal cut set/sequence generation for dynamic fault trees. In: Reliability and Maintainability, 2004 Annual Symposium - RAMS. (2004)
5. Klamt, S., Gilles, E.D.: Minimal cut sets in biochemical reaction networks. Bioinformatics **20**(2) (2004) 226–234
6. Samaga, R., Von Kamp, A., Klamt, S.: Computing combinatorial intervention strategies and failure modes in signaling networks. Journal of computational biology : a journal of computational molecular cell biology **17**(1) (Jan 2010) 39–53
7. Tarjan, R.: Depth-first search and linear graph algorithms. SIAM Journal on Computing **1**(2) (1972) 146–160
8. Schaefer, C.F., Anthony, K., Krupa, S., Buchoff, J., Day, M., Hannay, T., Buetow, K.H.: PID: The Pathway Interaction Database. Nucleic Acids Res. **37** (2009) D674–9
9. Paulevé, L., Magnin, M., Roux, O.: Refining dynamics of gene regulatory networks in a stochastic π-calculus framework. In: Transactions on Computational Systems Biology XIII. Volume 6575 of Lecture Notes in Comp Sci. Springer (2011) 171–191
10. Moustakas, A., Heldin, C.H.: Signaling networks guiding epithelial-mesenchymal transitions during embryogenesis and cancer progression. Cancer Sci. **98**(10) (Oct 2007) 1512–1520
11. Drabsch, Y., Ten Dijke, P.: TGF-β signalling and its role in cancer progression and metastasis. Cancer Metastasis Rev. **31**(3-4) (Dec 2012) 553–568
12. Minato, S.: Zero-suppressed bdds for set manipulation in combinatorial problems. In: Design Automation, 1993. 30th Conference on. (june 1993) 272 – 277