Characterization of Reachable Attractors Using Petri Net Unfoldings

Thomas Chatain[1], Stefan Haar[1], Loïg Jezequel[1], Loïc Paulevé[2,3], and Stefan Schwoon[1]

[1] LSV, ENS Cachan, INRIA, CNRS, France
[2] CNRS & Laboratoire de Recherche en Informatique UMR CNRS 8623
Université Paris-Sud, 91405 Orsay Cedex, France
[3] Inria Saclay - Île de France, team AMIB, Palaiseau, France

Abstract. Attractors of network dynamics represent the long-term behaviours of the modelled system. Their characterization is therefore crucial for understanding the response and differentiation capabilities of a dynamical system. In the scope of qualitative models of interaction networks, the computation of attractors reachable from a given state of the network faces combinatorial issues due to the state space explosion.
In this paper, we present a new algorithm that exploits the concurrency between transitions of parallel acting components in order to reduce the search space. The algorithm relies on Petri net unfoldings that can be used to compute a compact representation of the dynamics. We illustrate the applicability of the algorithm with Petri net models of cell signalling and regulation networks, Boolean and multi-valued. The proposed approach aims at being complementary to existing methods for deriving the attractors of Boolean models, while being generic since it applies to any safe Petri net.

Keywords: dynamical systems, attractors, concurrency, qualitative models, biological networks

1 Introduction

Living cells embed multiple regulation processes that lead to several emerging phenotypes such as cell differentiation, division, or response to environmental stress or signals. A large part of these processes are often represented as interaction networks (e.g., signalling networks, gene regulation networks) that describe the influences between numerous entities (genes, RNA, proteins). The global dynamics of such networks can then be captured using qualitative modelling frameworks, such as Boolean or discrete networks, that describe the possible transitions between the qualitative states of the system.

In the landscape of dynamics of a network model, one can distinguish between the transient and long-run dynamics, the latter being our focus in this article. In qualitative models, the long-run dynamics are referred to as *attractors*, and are formally defined as the Bottom Strongly Connected Components (BSCCs) of the transition graph whose nodes are the global states of the network, and directed

edges are the possible direct transitions between those states. One typically distinguishes between two kind of attractors: the fixed points, that are the states from which no further transition is possible; and the cyclic attractors, that are a set of states that can be visited infinitely often.

Characterizing the attractors of network dynamics is key for capturing the potential adaptation and differentiation processes the cell can undergo. In particular, one could verify, from a given state of the network, if the dynamics always converges toward a unique attractor or may diverge towards different attractors. The former indicates a deterministic long-term behaviour, whereas the latter suggests an indeterministic differentiation, potentially controlled by additional mechanisms not captured by the level of abstraction of the model.

In practice, given a qualitative model of a network, the computation of the attractors reachable from one (set of) states can become very expensive as the size of the network grows. The naive approach consisting in generating the transition graph and computing the BSSCs suffers from the combinatorial explosion of the state space (exponential with the number of components of the network) and the explosion of the number of transitions.

A part of the combinatorial explosion of dynamics is due to the concurrency between the asynchronous transitions: in a given state, several transitions may be independently fired, which results in numerous redundant interleavings of transitions in the concrete transition space.

Contribution. In this paper, we propose a new algorithm for characterizing all the attractors that are reachable from a given initial state in a qualitative model expressed with *safe* Petri nets [18], a broad class of nets which encompasses asynchronous Boolean or multi-valued networks. Our algorithm exploits the *unfoldings* of safe Petri net in order to reduce the size of the state space to explore.

Petri net unfoldings [7,8] aim at representing the state space rechable from an initial state by exploiting concurrency between transitions to prune redundant interleavings of these transitions. To our knowledge this is the first algorithm for computing all the reachable attractors which relies on unfolding structures. Our algorithm is applicable to any safe Petri net.

Whereas experiments on particular cases of biological networks show room for improvement, such a technique exploiting concurrency is foremost complementary to existing algorithms (which ignore this dynamical feature) and therefore appeals for designing combinations of techniques to make tractable the analysis of very large networks.

Related work. In the scope of Boolean networks, there has been numerous work to link the topology of the network (the interaction graph, giving signed relations between the components) with the fixed points - resulting in bounds or characterization of a subset of fixed points (e.g., [1,23,22]); and with cyclic attractors, e.g., [24,17]. While the full characterization of the fixed points of Boolean/multi-valued networks can be quite efficient for large networks [20,21,12], the complete characterization of cyclic attractors is still a challenging task due to the combinatoric explosition of the state space to explore. Symbolic representation of

the state space using binary decision diagrams has been used by [11] to characterize attractors in synchronous and asynchronous Boolean networks. The relationships between attractors in synchronous and asynchronous settings has then been exploited in [3] to speed the exploration of all possible attractors in Boolean networks, as well as Boolean network reduction techniques in [29]. Approximate methods are also largely considered, such as in [30], which selectively explore appropriate regions of the state space to derive a subset of cyclic attractors of the global network dynamics.

In this paper, we will focus on the use of unfolding to compute *finite complete prefixes* [14] of safe Petri nets. Finite complete prefixes contain all the reachable markings in a compact representation (the prefix is always smaller than the reachability graph). Unfoldings are very well suited to capture concurrent system dynamics, and can be efficient for reachability analysis [9], for instance.

Outline. In Sect. 2, we give a formal definition of (safe) Petri nets and their attractors, and introduce a running example. In Sect. 3, we present the unfolding of safe Petri nets. In Sect. 4, we detail our new algorithm to derive the attractors of a safe Petri net using its unfolding. Finally, implementation and experimations on Petri net models of biological networks are discussed in Sect. 5.

2 Petri Nets and Attractors

A Petri net is a bipartite graph where nodes are either places or transitions. In this paper, we consider only *safe* Petri nets where a place is either active or inactive (in oppositon to general Petri nets where each place can receive an arbitrary number of tokens, safe Petri nets allow at most one token per place). The set of active places form the state, or marking, of the net. A transition is said enabled if all the places that are parents of the transition are active. In the semantics, the *firing* of a transition makes inactive the parent places and then makes active the children places, modifying the current marking of the net.

Formally, a *(safe) Petri net* is a tuple $\mathcal{N} = \langle P, T, F, M_0 \rangle$ where P and T are sets of *nodes* (called *places* and *transitions* respectively), and $F \subseteq (P \times T) \cup (T \times P)$ is a *flow relation* (whose elements are called *arcs*). A subset $M \subseteq P$ of the places is called a marking, and $M_0 = \{p_0^1, \ldots, p_0^n\}$ is a distinguished *initial marking*. For any node $x \in P \cup T$, we call *pre-set* of x the set ${}^\bullet x = \{y \in P \cup T \mid (y, x) \in F\}$ and *post-set* of x the set $x^\bullet = \{y \in P \cup T \mid (x, y) \in F\}$.

A transition $t \in T$ is *enabled* at a marking M if and only if ${}^\bullet t \subseteq M$. Then t can *fire*, leading to the new marking $M' = (M \setminus {}^\bullet t) \cup t^\bullet$. We write $M \xrightarrow{t} M'$. A *firing sequence* is a (finite or infinite) word $w = t_1 t_2 \ldots$ over T such that there exist markings $M_1, M_2, \ldots$ such that $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} M_2 \ldots$ For any such firing sequence w, the markings $M_1, M_2, \ldots$ are called *reachable markings*.

The Petri nets we consider are said to be *safe* because we will assume that any reachable marking M is such that for any $t \in T$ that can fire from M leading to M', the following property holds: $\forall p \in M \cap M'$, $p \in {}^\bullet t \cap t^\bullet \vee p \notin {}^\bullet t \cup t^\bullet$.

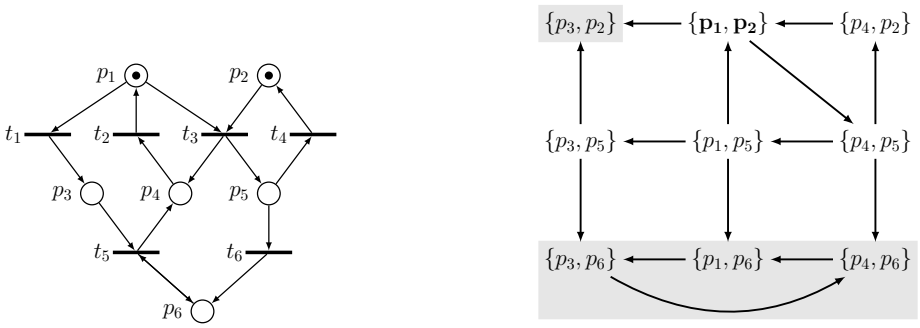

Fig. 1. A safe Petri net (left) and the corresponding marking graph (right) - the initial marking is in bold.

Figure 1 (left) shows an example of a safe Petri net. The places are represented by circles and the transitions by horizontal lines (each one with a label identifying it). The arrows represent the arcs. The initial marking is represented by dots (or tokens) in the marked places.

From an initial marking of the net, one can recursively derive all possible transitions and reachable markings, resulting in the *marking graph* (Def. 1). The marking graph is always finite in the case of safe Petri nets. The attractors reachable from the initial marking of the net can then be fully characterized by the bottom strongly connected components of the marking graph (Def. 2).

Definition 1 (Marking graph). *The marking graph of a Petri Net $\mathcal{N}$ is a directed graph $\mathcal{G} = (V, A)$ such that V is the set of all reachable markings (obtained from all the possible firing sequences) and $A \subseteq V \times V$ is such that $(M, M') \in A$ if and only if $M \xrightarrow{t} M'$ for some $t \in T$.*

Definition 2 (Attractors). *An* attractor *is a bottom strongly connected component of $\mathcal{G}$, that is a set $\mathcal{A}$ of markings such that either $\mathcal{A} = \{M\}$ and no transition is enabled from M; or for every $M \in \mathcal{A}$, the set of markings reachable from M is precisely $\mathcal{A}$.*

Figure 1 (right) represents the marking graph of the Petri net of Figure 1 (left). The two attractors of the Petri net of Figure 1 (left) are evidenced by the grey parts of its marking graph in Figure 1 (right).

3 Unfoldings

In this section, we explain the basics of Petri net unfoldings. A more extensive treatment of the theory explained here can be found, e.g., in [7]. Roughly speaking, the unfolding of a Petri net $\mathcal{N}$ is an "acyclic" Petri net $\mathcal{U}$ that has the same behaviours as $\mathcal{N}$ (modulo homomorphism). In general, $\mathcal{U}$ is an infinite net, but if $\mathcal{N}$ is safe, then it is possible [16] to compute a finite prefix $\mathcal{P}$

of $\mathcal{U}$ that is "complete" in the sense that every reachable marking of $\mathcal{N}$ has a reachable counterpart in $\mathcal{P}$. Thus, $\mathcal{P}$ represents the set of reachable markings of $\mathcal{N}$. Figure 2 shows a finite complete prefix of the unfolding of the Petri net of Figure 1.

In principle, the set of reachable markings can also be computed by constructing the marking graph (see Definition 1). However, the marking graph suffers from combinatorial explosion due to concurrency. For instance, suppose that $\mathcal{N}$ simply contains n independent concurrent actions. Then the only attractor of the net is reached by executing all n actions in any arbitrary order. However, the marking graph will uselessly explore all $n!$ different schedules for executing them, and all 2^n intermediate markings.

Research into concurrent systems has produced a number of solutions to alleviate the problem of combinatorial explosion due to concurrency (and eventually other sources). In [16], McMillan first proposed the use of finite unfolding prefixes. Esparza et al [8] later improved this solution. For instance, the unfolding of the previous example with n independent actions is simply of size $\mathcal{O}(n)$. With respect to the marking graph, an unfolding represents a time-space tradeoff: in general, a complete unfolding prefix $\mathcal{P}$ is much smaller than the marking graph of $\mathcal{N}$, but the problem whether a marking M of $\mathcal{N}$ is reachable, given $\mathcal{P}$, is NP-complete. However, this tradeoff is usually favourable [9].

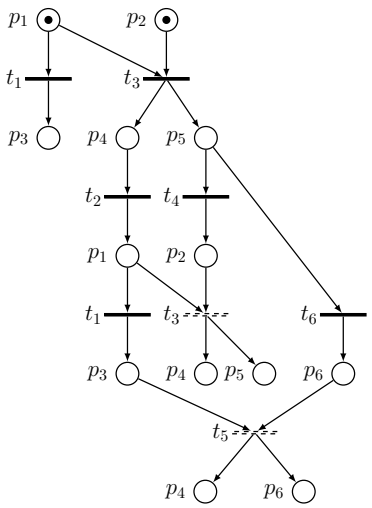

Fig. 2. A finite complete prefix of the unfolding of the Petri net of Figure 1. Dashed events are flagged as *cut-offs*: the unfolding procedure does not continue beyond them.

We now give some technical definitions to introduce unfoldings formally.

Definition 3 (Causality, conflict, concurrency). *Let $\mathcal{N} = \langle P, T, F, M_0 \rangle$ be a net and $x, y \in P \cup T$ two nodes of $\mathcal{N}$. We say that x is a* causal predecessor *of y, noted $x < y$, if there exists a non-empty path of arcs from x to y. We note $x \leq y$ if $x < y$ or $x = y$. If $x \leq y$ or $y \leq x$, then x and y are said to be* causally related*. x and y are* in conflict*, noted $x \# y$, if there exist $u, v \in T$ such that $u \leq x$, $v \leq y$, and ${}^\bullet u \cap {}^\bullet v \neq \emptyset$. We call x and y* concurrent*, noted $x \mathrel{co} y$, if they are neither causally related nor in conflict.*

As we said before, an unfolding is an "acyclic" version of a net $\mathcal{N}$. This notion of acyclicity is captured by Definition 4.

Definition 4 (Occurrence net). *Let $\mathcal{N} = \langle P, T, F, M_0 \rangle$ be a Petri net. We say that $\mathcal{N}$ is an* occurrence net *if it satisfies the following properties:*

1. *The causality relation* $<$ *is acyclic;*
2. $|{}^\bullet p| \leq 1$ *for all places* p*, and* $M(p) = 1$ *iff* $|{}^\bullet p| = 0$*;*
3. *for every transition* t*,* $t \# t$ *does not hold, and* $\{x \mid x \leq t\}$ *is finite.*

As is convention in the unfolding literature, we shall refer to the places of an occurrence net as *conditions* and to its transitions as *events*. Due to the structural constraints, the firing sequences of occurrence nets have special properties: if some condition c is marked during a run, then the token on c was either present initially or produced by one particular event (the single event in ${}^\bullet c$); moreover, once the token on c is consumed, it can never be replaced by another token, due to the acyclicity constraint on $<$.

Definition 5 (Configuration, cut). *Let* $\mathcal{N} = \langle C, E, F, M\rangle$ *be an occurrence net. A set* $\mathcal{C} \subseteq E$ *is called* configuration *of* $\mathcal{N}$ *if (i)* $\mathcal{C}$ *is* causally closed*, i.e. for all* $e, e' \in E$ *with* $e' < e$*, if* $e \in \mathcal{C}$ *then* $e' \in \mathcal{C}$*; and (ii)* $\mathcal{C}$ *is conflict-free, i.e. if* $e, e' \in \mathcal{C}$*, then* $\neg(e \# e')$*. The* cut *of* $\mathcal{C}$*, denoted* $Cut(\mathcal{C})$*, is the set of conditions* $(M \cup \mathcal{C}^\bullet) \setminus {}^\bullet\mathcal{C}$*.*

Intuitively, a configuration is a set of events that can fire during a firing sequence of $\mathcal{N}$, and its cut is the set of conditions marked after that firing sequence.

We can now define the notion of unfoldings. Let $\mathcal{N} = \langle P, T, F, M_0\rangle$ be a safe Petri net. The unfolding $\mathcal{U} = \langle C, E, G, M_0'\rangle$ of $\mathcal{N}$ is an (infinite) occurrence net (equipped with a homomorphism h) such that the firing sequences and reachable markings of $\mathcal{U}$ are exactly the firing sequences and reachable markings of $\mathcal{N}$ (modulo h). $\mathcal{U}$ can be inductively constructed as follows:

1. The conditions C are a subset of $(E \cup \{\bot\}) \times P$. For a condition $c = \langle x, p\rangle$, we will have $x = \bot$ iff $x \in M_0'$; otherwise x is the singleton event in ${}^\bullet c$. Moreover, $h(c) = p$. The initial marking M_0' contains exactly one condition $\langle \bot, p\rangle$ for each initially marked place p of $\mathcal{N}$.
2. The events of E are a subset of $2^C \times T$. More precisely, we have an event $e = \langle C', t\rangle$ for every set $C' \subseteq C$ such that $c \mathrel{co} c'$ holds for all $c, c' \in C'$ and $\{\, h(c) \mid c \in C' \,\} = {}^\bullet t$. In this case, we add edges $\langle c, e\rangle$ for each $c \in C'$ (i.e. ${}^\bullet e = C'$), we set $h(e) = t$, and for each $p \in t^\bullet$, we add to C a condition $c = \langle e, p\rangle$, connected by an edge $\langle e, c\rangle$.

Intuitively, a condition $\langle x, p\rangle$ represents the possibility of putting a token onto place p through a particular firing sequence, while an event $\langle C', t\rangle$ represents a possibility of firing transition t in a particular context.

Recall that a configuration $\mathcal{C}$ of $\mathcal{U}$ represents a possible firing sequence, whose resulting marking corresponds, due to the construction of $\mathcal{U}$, to a reachable marking of $\mathcal{N}$. This marking is defined as $Mark(\mathcal{C}) := \{\, h(c) \mid c \in Cut(\mathcal{C}) \,\}$. Since $\mathcal{U}$ is infinite in general, we are interested in computing an initial portion of it (a *prefix*) that completely characterizes the behaviour of $\mathcal{N}$.

Definition 6 (complete prefix). *Let* $\mathcal{N} = \langle P, T, F, M_0\rangle$ *be a safe Petri net and* $\mathcal{U} = \langle C, E, G, M_0'\rangle$ *its unfolding. A finite occurrence net* $\mathcal{P} = \langle C', E', G', M_0'\rangle$

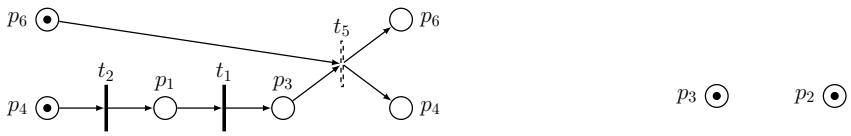

Fig. 3. Two attractors of the Petri net of Figure 1 represented as finite complete prefixes $\mathcal{U}_{\{p_4,p_6\}}$ (on the left) and $\mathcal{U}_{\{p_3,p_2\}}$ (on the right).

is said to be a prefix *of* $\mathcal{U}$ *if* $E' \subseteq E$ *is causally closed,* $C' = M_0' \cup E'^{\bullet}$*, and* G' *is the restriction of* G *to* C' *and* E'*. A prefix* $\mathcal{P}$ *is said to be* complete *if for every reachable marking* M *of* $\mathcal{N}$ *there exists a configuration* $\mathcal{C}$ *of* $\mathcal{P}$ *such that (i)* $Mark(\mathcal{C}) = M$*, and (ii) for each transition* $t \in T$ *enabled in* M*, there is an event* $\langle C'', t\rangle \in E'$ *enabled in* $Cut(\mathcal{C})$*.*

It is known [16,8] that the construction of such a complete prefix is indeed possible, and efficient tools [26,13] exist for this purpose. The precise details of this construction are out of scope for this paper; for what follows it suffices to know that it essentially follows the construction of $\mathcal{U}$ outlined above but that certain events are flagged as *cut-offs* when they do not "contribute any new reachable markings" (these events are represented by dashed lines in Figure 2). The construction then does not continue "beyond" such a cut-off event.

4 Extracting attractors from unfoldings

Our method for finding the attractors of a Petri net $\mathcal{N}$ uses unfoldings in two steps: first to find a set of markings which intersects all the attractors, second to output the attractors as a set of finite complete prefixes.

4.1 Representation of attractors as finite complete prefixes

We first remark that finite complete prefixes of unfoldings are particularly well suited for the representation of attractors. In fact, every attractor $\mathcal{A}$ is a set of states which form a maximal strongly connected component of the marking graph of $\mathcal{N}$. For every marking M in $\mathcal{A}$, the attractor $\mathcal{A}$ is precisely the set of markings reachable from M. Hence it can be compactly represented as a finite complete prefix of the unfolding of the Petri net $\mathcal{N}$ initialized at M. Denote this prefix $\mathcal{U}_M$: the markings associated to the configurations of $\mathcal{U}_M$ are precisely those of the attractor, moreover the prefix shows the dynamics of the net while in the attractor. Last, the size of $\mathcal{U}_M$ (as number of non cut-off events) can be up to exponentially smaller (in case of highly concurrent behaviour) than the number of markings in the attractor, and never exceeds it.

Figure 3 shows two attractors of the Petri net $\mathcal{N}$ of Figure 1 represented as finite complete prefixes. The one on the left, $\mathcal{U}_{\{p_4,p_6\}}$, represents the attractor containing the marking $\{p_4, p_6\}$. The one on the right represents the attractor made of the single marking $\{p_3, p_2\}$ which is a deadlock.

4.2 Maximal configurations and attractors

We have shown the interest of prefixes for representing attractors. What we need is a way to find a set $\mathcal{M}$ of markings of the Petri net $\mathcal{N}$ which contains one marking per attractor. Given such $\mathcal{M}$, the set $\{\mathcal{U}_M \mid M \in \mathcal{M}\}$ gives a complete characterization of the attractors of $\mathcal{N}$.

Now we show that the desired set $\mathcal{M}$ of markings can be obtained from the maximal configurations of a finite complete prefix of the unfolding of $\mathcal{N}$.

Definition 7 (maximal configuration). *A configuration of a prefix is called* maximal *if no other event of the prefix can be added to the configuration. Equivalently, the configuration is a deadlock of the prefix viewed as a Petri net.*

For example, in the prefix shown on Figure 2, the configuration corresponding to the firing sequence t_3t_2 is not maximal because it can be extended, for instance by t_4 and the other event labeled t_3, yielding this time a maximal configuration, which reaches the marking $\{p_4, p_5\}$. Notice that this marking is not a deadlock in the original Petri net, yet the configuration is maximal in the prefix.

Property 1. Let $\mathcal{N}$ be a Petri net and $\mathcal{U}$ a finite complete prefix of its unfolding. For every attractor $\mathcal{A}$ of $\mathcal{N}$, there exists a maximal configuration of $\mathcal{U}$ whose associated marking belongs to $\mathcal{A}$.

Proof. Choose a marking M in $\mathcal{A}$. Because $\mathcal{U}$ is complete, it has a configuration $\mathcal{C}$ whose associated marking is M. Now because $\mathcal{U}$ is finite, it has a maximal configuration $\mathcal{C}'$ which extends $\mathcal{C}$. The marking M' associated to $\mathcal{C}'$ is reachable from M, therefore it is also in $\mathcal{A}$.

The prefix shown on Figure 2 has four maximal configurations. One is obtained after firing only t_1; its associated marking is the deadlock $\{p_3, p_2\}$. This marking is associated to another maximal configuration: the one obtained by firing t_3 and then t_2t_1 concurrently with t_4. The third maximal configurations is obtained by firing t_3, then t_2 and t_4 concurrently, and then t_3 again; it reaches the marking $\{p_4, p_5\}$. Finally, one can fire t_3, then t_2t_1 and t_6 concurrently, and then t_5; it reaches the marking $\{p_4, p_6\}$.

One can check that every attractor has a marking in this set: the deadlock $\{p_3, p_2\}$ is represented twice; the marking $\{p_4, p_6\}$ represents the attractor $\{\{p_3, p_6\}, \{p_1, p_6\}, \{p_4, p_6\}\}$. The marking $\{p_4, p_5\}$, also associated to a maximal configuration of $\mathcal{U}$, is not in an attractor.

4.3 Algorithm

Property 1 allows one to use finite complete prefixes to identify attractors: the set $\mathcal{M}_{\max}$ of markings corresponding to maximal configurations intersects all the attractors. But not all the markings in $\mathcal{M}_{\max}$ belong to an attractor. Also an attractor may contain several markings in $\mathcal{M}_{\max}$.

In order to characterize the attractors of a safe Petri net $\mathcal{N}$, we filter the set $\mathcal{M}_{\max}$ and keep only one marking per attractor. The idea is to remove iteratively the markings from which another marking of the set is reachable. The reachability checking is done again using unfoldings.

The algorithm is the following:

1. Compute a finite complete prefix $\mathcal{U}$ of the unfolding of $\mathcal{N}$.
2. Initialize $\mathcal{M}$ to the set $\mathcal{M}_{\max}$ of markings corresponding to maximal configurations of $\mathcal{U}$.
3. Initialize the set of attractors to $\emptyset$.
4. Loop for M in $\mathcal{M}$
   - Compute a finite complete prefix $\mathcal{U}_M$ of the net $\mathcal{N}$ initialized at M.
   - If a marking $M' \in \mathcal{M}$ other than M is reachable from M (the reachability check is done using $\mathcal{U}_M$),
     Then remove M from $\mathcal{M}$,
     Else add $\mathcal{U}_M$ to the set of attractors.
5. Output the set of attractors.

Termination of the algorithm is straightforward. We prove that at every step of the algorithm, the set $\mathcal{M}$ intersects all the attractors. This property is preserved when we remove a marking M from $\mathcal{M}$ because, if M is in an attractor $\mathcal{A}$, then the marking $M' \in \mathcal{M}$ reachable from M is also in $\mathcal{A}$. Notice also that, if M is not in an attractor, then at least one attractor $\mathcal{A}$ is reachable from M; and because $\mathcal{M} \cap \mathcal{A} \neq \emptyset$, $\mathcal{M}$ contains a marking $M' \in \mathcal{A}$ which is reachable from M. This ensures that $\mathcal{U}_M$ is added to the set of attractors iff M is in an attractor $\mathcal{A}$ and $\mathcal{M} \cap \mathcal{A} = \{M\}$.

4.4 Illustration on the running example

For our running example the set $\mathcal{M}$ is initialized to $\{\{p_3, p_2\}, \{p_4, p_5\}, \{p_4, p_6\}\}$. The algorithm computes the prefix $\mathcal{U}_M$ for every $M \in \mathcal{M}$, but outputs only $\mathcal{U}_{\{p_3,p_2\}}$ and $\mathcal{U}_{\{p_4,p_6\}}$, pictured in Figure 3. $\mathcal{U}_{\{p_4,p_5\}}$ is dropped because $\{p_4, p_6\}$ is reachable from $\{p_4, p_5\}$.

5 Implementation and Experimental Results

In order to test the applicability of our approach, we implemented a prototype of the algorithm described above using `Mole`[26] for computing the complete prefixes, and `Minisat`[6] for extracting the maximal configurations[4].

We applied our algorithm for the identification of attractors of three qualitative models of biological networks taken from the literature. In all cases, we applied a transformation to safe Petri net (Appendix A) that consists in having one place for each qualitative level of each component of the network, and

[4] Executables, scripts, and Petri net models are available for Linux 64bits at http://loicpauleve.name/cmsb2014.tbz2

transitions corresponding to the asynchronous semantics, i.e., each transition will actually change the level of only one component. By construction, the places corresponding to the levels of each components are mutually exclusive. The initial marking may then correspond either to a single state of the qualitative model, or to several possible initial states by adding transitions that non-deterministically select the initial state for some components (Sect. A.2). In this latter encoding, the returned attractors are the attractors reachable from at least one of the possible initial state of the qualitative model.

Model	Nb. nodes	Nb. max. conf.	Nb. attractors
Lambda switch	4	15	2
Cell cycle	10	12	1
ERBB (1)	20	301	1
ERBB (2)	20	302	2
VPC C. elegans	88	1240	1

Table 1. Results of the attractors characterization using Petri net unfoldings. For each model (which includes the initial state), we give the number of maximal configurations and the number of attractors reachable from the initial state.

Tab. 1 sums up the results of computing the finite complete prefixes on the following regulatory networks: a multi-valued model of the lambda switch [27], a Boolean model of he mammalian cell cycle [10], a Boolean model of the ERBB receptor [25], and a multi-valued model of fate determination in the Vulval Precursor Cells (VPC) in C. elegans [28]. For the ERBB model, two different initial settings have been tested: (1) when EGF is active; (2) when either EGF is active or inactive. For the other models, the initial state is the level 0 for all the components. The execution times are in the order of a fraction of a second for the two first models; in the order of a few seconds for ERBB; and around 15 minutes for the VPC model. For the latter, we note that starting from a different initial state leads to a combinatoric explosion of the complete prefix, showing room for improvement to handle large model in general. It is difficult to compare with other existing tools as most of them handle only Boolean networks and do not support the search from a given initial state. GINsim [19] also provides attractors computations from a given initial state, but it relies on explicit state transition graph computation, which is always larger than a complete prefix (Section 3). For instance on the VPC example, GINsim has been stopped after one hour.

6 Discussion

We presented a new algorithm for computing all the attractors reachable from a given state in the general class of safe Petri nets, i.e., Petri nets having at most one token in each place. This class includes Boolean and multi-valued networks that are typically used to model the qualitative dynamics of biological networks.

Our approach relies on Petri net unfoldings that natively take into account the concurrency between transitions to produce a compact representation of the reachable states. Then, we use the notion of *maximal configuration* to derive, from the computed unfolding, a set of states that includes at least one state of each reachable attractor. This set of maximal configuration is then filtered to output exactly one state per reachable attractors. The identification of attractors is complete in the sense that all the attractors reachable from the supplied markings are detected, including fixed points and cyclic and complex attractors.

By the use of Petri unfoldings, we aim at reducing the complexity of exploring the full reachable space by inherently pruning redundant transitions due to some interleaving of concurrent transitions. We applied a prototype implementation of the algorithm to four biological networks ranging from four to eighty interacting components, either multi-valued or Boolean.

The unfolding technique mentioned in this paper is generic to any safe Petri net. This indicates several directions for improving its computation (including the extraction of maximal configurations) in the particular case of biological interaction networks, such as the use of contextual Petri nets [2], merge processes [15], unravelings [4], and the use of the network topology and static analysis to prune non-necessary transitions and decompose the detection of attractors.

References

1. J. Aracena. Maximum number of fixed points in regulatory boolean networks. *Bull. Math. Biol.*, 70(5):1398–1409, 2008.
2. P. Baldan, A. Bruni, A. Corradini, B. König, C. Rodríguez, and S. Schwoon. Efficient unfolding of contextual Petri nets. *TCS*, 449:2–22, 2012.
3. N. Berntenis and M. Ebeling. Detection of attractors of large boolean networks via exhaustive enumeration of appropriate subspaces of the state space. *BMC Bioinformatics*, 14(1):361, 2013.
4. G. Casu and G. M. Pinna. Flow unfolding of safe nets. In *Petri Nets*, 2014.
5. C. Chaouiya, A. Naldi, E. Remy, and D. Thieffry. Petri net representation of multi-valued logical regulatory graphs. *Natural Computing*, 10(2):727–750, 2011.
6. N. Eén and N. Sörensson. An extensible sat-solver. In *Theory and applications of satisfiability testing*, pages 502–518. Springer, 2004.
7. J. Esparza and K. Heljanko. *Unfoldings – A Partial-Order Approach to Model Checking*. Springer, 2008.
8. J. Esparza, S. Römer, and W. Vogler. An improvement of McMillan's unfolding algorithm. *FMSD*, 20:285–310, 2002.
9. J. Esparza and C. Schröter. Unfolding based algorithms for the reachability problem. *Fund. Inf.*, 47(3-4):231–245, 2001.
10. A. Faure, A. Naldi, C. Chaouiya, and D. Thieffry. Dynamical analysis of a generic Boolean model for the control of the mammalian cell cycle. *Bioinformatics*, 22(14):124–131, 2006.
11. A. Garg, A. Di Cara, I. Xenarios, L. Mendoza, and G. De Micheli. Synchronous versus asynchronous modeling of gene regulatory networks. *Bioinformatics*, 24(17):1917–1925, 2008.

12. F. Hinkelmann, M. Brandon, B. Guang, R. McNeill, G. Blekherman, A. Veliz-Cuba, and R. Laubenbacher. ADAM: Analysis of discrete models of biological systems using computer algebra. *BMC Bioinformatics*, 12(1):295, 2011.
13. V. Khomenko. Punf. http://homepages.cs.ncl.ac.uk/victor.khomenko/tools/punf/.
14. V. Khomenko, M. Koutny, and W. Vogler. Canonical prefixes of Petri net unfoldings. *Acta Inf.*, 40(2):95–118, 2003.
15. V. Khomenko and A. Mokhov. An algorithm for direct construction of complete merged processes. In *Petri Nets*, pages 89–108, 2011.
16. K. L. McMillan. Using unfoldings to avoid the state explosion problem inthe verification of asynchronous circuits. In *CAV*, pages 164–177, 1992.
17. T. Melliti, M. Noual, D. Regnault, S. Sené, and J. Sobieraj. Full characterization of attractors for two intersected asynchronous boolean automata cycles. *CoRR*, abs/1310.5747, 2013.
18. T. Murata. Petri nets: Properties, analysis and applications. *Proc. of the IEEE*, 77(4):541–580, 1989.
19. A. Naldi, D. Berenguier, A. Faur, F. Lopez, D. Thieffry, and C. Chaouiya. Logical modelling of regulatory networks with GINsim. *Biosystems*, 97(2):134 – 139, 2009.
20. A. Naldi, D. Thieffry, and C. Chaouiya. Decision diagrams for the representation and analysis of logical models of genetic networks. In *CMSB*, pages 233–247, 2007.
21. L. Paulevé, M. Magnin, and O. Roux. Refining dynamics of gene regulatory networks in a stochastic π-calculus framework. *TCSB*, 6575:171–191, 2011.
22. L. Paulevé and A. Richard. Topological Fixed Points in Boolean Networks. *C. R. Acad. Sci. - Series I - Mathematics*, 348(15-16):825 – 828, 2010.
23. A. Richard. Positive circuits and maximal number of fixed points in discrete dynamical systems. *Discrete Appl. Math.*, 157(15):3281 – 3288, 2009.
24. A. Richard. Negative circuits and sustained oscillations in asynchronous automata networks. *Adv. in Appl. Math.*, 44(4):378 – 392, 2010.
25. O. Sahin, H. Frohlich, C. Lobke, U. Korf, S. Burmester, M. Majety, J. Mattern, I. Schupp, C. Chaouiya, D. Thieffry, A. Poustka, S. Wiemann, T. Beissbarth, and D. Arlt. Modeling ERBB receptor-regulated G1/S transition to find novel targets for de novo trastuzumab resistance. *BMC Systems Biology*, 3(1), 2009.
26. S. Schwoon. Mole. http://www.lsv.ens-cachan.fr/ schwoon/tools/mole/.
27. D. Thieffry and R. Thomas. Dynamical behaviour of biological regulatory networks – II. Immunity control in bacteriophage lambda. *Bull. Math. Biol.*, 57:277–297, 1995.
28. N. Weinstein and L. Mendoza. A network model for the specification of vulval precursor cells and cell fusion control in caenorhabditis elegans. *Frontiers in Genetics*, 4(112), 2013.
29. J. G. T. Zañudo and R. Albert. An effective network reduction approach to find the dynamical repertoire of discrete dynamic networks. *Chaos*, 23:025111, 2013.
30. D. Zheng, G. Yang, X. Li, Z. Wang, F. Liu, and L. He. An efficient algorithm for computing attractors of synchronous and asynchronous boolean networks. *PLoS ONE*, 8(4):e60593, 2013.

A Encoding asynchronous discrete networks with safe Petri Nets

A.1 Encoding with one initial state

In literature, Boolean and multi-valued networks modelling dynamics of biological influence networks are typically represented by functions associating for

each component the levels towards which it evolves with respect to each possible level of its regulators. In order to encode their asynchronous dynamics in Petri nets, one need to have a transition-centered representation, instead of a function-centered. Informally, this can be achieved by having one place per possible level of each component (we note i_u the place corresponding to the level u of component i), and listing the conditions for moving a token from i_u to i_{u+1} and i_{u-1}. Such conditions can typically be built from the expression of the discrete functions of the network. Our encoding always results in safe Petri nets, which makes it different from [5] which relies on more advanced Petri nets semantics (multiple tokens on places and weighted arcs),

A Discrete Network gathers a finite number of components $i \in \{1, \cdots, n\}$ having a discrete finite domain $\mathbb{F}^i$ that we note $\mathbb{F}^i = \{0, \cdots, l_i\}$, l_i being the maximum level for the component i. For each component $i \in \{1, \cdots, n\}$, a map $f^i : \mathbb{F} \to \mathbb{F}^i$ is defined, where $\mathbb{F} = \mathbb{F}^1 \times \cdots \times \mathbb{F}^n$, giving the next value of the component with respect to the global state of the network. Typically f^i depends on a subset of components (its regulators) that we denote $\mathsf{dep}(f^i)$. In the case of Asynchronous Discrete Networks (ADN), a transition relation $\to_{ADN} \subseteq \mathbb{F} \times \mathbb{F}$ is defined such that $x \to_{ADN} x'$ if and only if there exists a unique $i \in \{1, \cdots, n\}$ such that $x'[i] = f^i(x)$ and $\forall j \in \{1, \cdots, n\}, j \neq i, x'[j] = x[j]$, i.e. one and only one component has been updated. This is formalised in Def. 8.

Definition 8 (Asynchronous Discrete Network (ADN)). *An ADN is defined by a couple* $(\mathbb{F}, \langle f^1, \ldots, f^n \rangle)$ *where* $\mathbb{F} = \mathbb{F}^1 \times \cdots \times \mathbb{F}^n$, *and* $\forall i \in \{1, \cdots, n\}$, $f^i : \mathbb{F} \to \mathbb{F}^i$ *with* $\mathbb{F}^i = \{0, \cdots, l_i\}$. *Given two states* $x, x' \in \mathbb{F}$, *the transition relation* $\to_{ADN}$ *is given by*

$$x \to_{ADN} x' \iff \exists i \in \{1, \cdots, n\}, f^i(x) = x'[i] \wedge \forall j \in \{1, \cdots, n\}, j \neq i, x[j] = x'[j] \ ,$$

where $x[i]$ *is the* i*-th component of* x. *We note* $\mathsf{dep}(f^i) \subseteq \{1, \cdots, n\}$ *the set of components on which the value of* f^i *depends:* $\forall x, x' \in \mathbb{F}$ *such that* $\forall j \in \mathsf{dep}(f^i), x[j] = x'[j]$, *necessarily* $f^i(x) = f^i(x')$.

In the scope of an ADN $(\mathbb{F}, \langle f^1, \ldots, f^n \rangle)$, we use $\mathsf{cond}(x)$ to map a state to the set of literals for the presence of the components at the corresponding state, e.g., $\mathsf{cond}(\langle 1, 0, 1 \rangle) = \{1_1, 2_0, 3_1\}$: $\mathsf{cond}(x) \stackrel{\Delta}{=} \{i_u \mid i \in \{1, \cdots, n\}, x[i] = u\}$. Given a component i at a state u, we note $\mathbf{conds}^i_{u+}$ and $\mathbf{conds}^i_{u-}$ the set of conditions where i can respectively increase or decrease. This set of conditions can be read as a disjunctive normal form expressing the possibility of the transition: $\mathbf{conds}^i_{u+} \stackrel{\Delta}{=} \mathsf{simplify}(\{\mathsf{cond}(x)|_{\mathsf{dep}(i)} \mid x \in \mathbb{F}, x[i] = u, f^i(x) > u\})$; $\mathbf{conds}^i_{u-} \stackrel{\Delta}{=} \mathsf{simplify}(\{\mathsf{cond}(x)|_{\mathsf{dep}(i)} \mid x \in \mathbb{F}, x[i] = u, f^i(x) < u\})$; where $\mathsf{simplify}$ is an operator to reduce the number of conditions, and $\mathsf{cond}(x)|_{\mathsf{dep}(i)}$ restricts the literals to those corresponding to components influencing i.

Finally, given an ADN $(\mathbb{F}, \langle f^1, \ldots, f^n \rangle)$ and an initial state $x_0 \in \mathbb{F}$, the corresponding safe Petri net is defined by (P, T, F, M_0) where $P = \{i_u \mid i \in \{1, \cdots, n\}, u \in \{0, \cdots, l_i\}\}$, $M_0 = \mathsf{cond}(x_0)$, and T and F are the smallest sets (w.r.t. inclusion) such that $\forall i \in \{1, \cdots, n\}$, $\forall u \in \{0, \cdots, l_i\}$, $\forall \Phi \in \mathbf{conds}^i_{u+} \cup$

$\mathbf{conds}^i_{u-}$, $\exists t \in T : {}^\bullet t = \Phi \cup \{a_i\} \wedge t^\bullet = (\Phi \setminus \{a_i\}) \cup \{a_j\}$. By construction, the Petri net is safe and for each $i \in \{1, \cdots, n\}$, the places $i_0, \cdots, i_{l_i}$ are mutually exclusive.

A.2 Encoding with multiple initial states

When studying the dynamics of a qualitative network, one may want to consider several initial states, chosen nondeterministically. Using the construction depicted in the previous section, one can encode this indeterministic choice by adding a place, initially marked, per (independent) indeterministic choice, and a transition per corresponding (local) state. Fig. 4 illustrate this construction with either an indeterministic initial state for one component, or for several components.

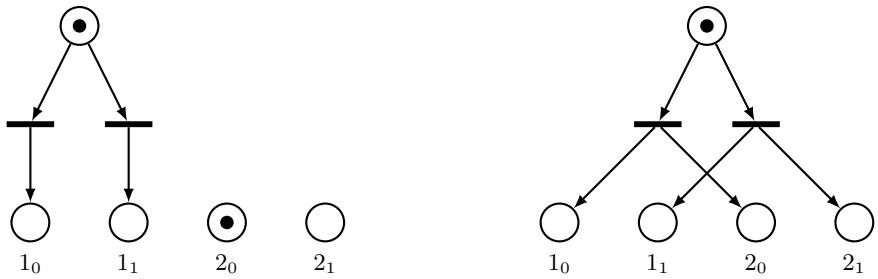

Fig. 4. (left) Encoding of the indeterministic choice between the initial state of 1; (right) indeterministic choice between the initial state of the couple 1 and 2: either $\langle 0, 0\rangle$ or $\langle 1, 1\rangle$.