Drawing the Line: Basin Boundaries in Safe Petri Nets

Stefan Haar, Loïc Paulevé, Stefan Schwoon

Drawing the Line: Basin Boundaries in Safe Petri Nets

Stefan Haar[1], Loïc Paulevé[2], and Stefan Schwoon[1]

[1] INRIA and LSV, CNRS & ENS Paris-Saclay, Université Paris-Saclay, France
[2] Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, Talence, France

Abstract. Attractors of network dynamics represent the long-term behaviours of the modelled system. Understanding the basin of an attractor, comprising all those states from which the evolution will eventually lead into that attractor, is therefore crucial for understanding the response and differentiation capabilities of a dynamical system. Building on our previous results [2] allowing to find attractors via Petri net Unfoldings, we exploit further the unfolding technique for a backward exploration of the state space, starting from a known attractor, and show how all strong or weak basins of attractions can be explicitly computed.
Keywords: dynamical systems, qualitative models, attractors, concurrency, biological networks, epigenetics, reprogramming

1 Introduction

Multistability is a central feature of models for biological systems. It is implied by many fundamental biological processes, such as cellular differentiation, cellular reprogramming, and cell fate decision.

In qualitative models such as Boolean and multivalued networks, multistability is tied to the notion of *attractors* and to their *basins*. Attractors are usually defined as the smallest subsets of states from which the system cannot escape. Then, basins refer to the states of the system which can reach a given attractor. One can distinguish two kinds of basins for an attractor A: the *weak* basin of A which gathers all the states that can reach A, but possibly others; and the *strong* basin of A which is the subset of the weak basin which cannot reach other attractors than A. The strong basin includes the attractor itself, and possibly other preceding states [15].

Understanding how the system switches from a weak to a strong basin is a recurrent question when analysing models of signalling and gene regulatory networks [5,19]. In [10], the authors provide a method for identifying the states in which one transition leads to losing the reachability of a given attractor (bifurcation transitions). Whereas the approach can still enumerate only the bifurcation transitions, it then loses the precious information of the contexts in which the transitions make the system bifurcate from the attractor. Thus, besides listing of the states on the boundary of a strong basins, the challenge resides in identifying the specific contexts and sequences of transitions leading to a strong basin.

Let us illustrate on the small automata network example showing bifurcation transitions reproduced in Fig. 1. The model gathers 3 automata a, b, and c with respectively 3, 2, and 3 local states. The automata start in the the state in gray, then can undergo transitions (fully) asynchronously. Transitions labeled with the local states of other automata can only be taken whenever the referenced automata are in these states. This results in the transition graph of Fig. 2. Two attractors are reachable: the fixpoint (i.e. singleton attractor) $\langle a_2, b_1, c_0 \rangle$, and the cyclic attractor where c is fixed to 2, and a and b oscillate between 0 and 1. The states in gray in Fig. 2 form the weak basin of the fixpoint$\langle a_2, b_1, c_0 \rangle$. In specific states of this basin, firing the transition t_8, which is the transition of c from 1 to 2 makes the system lose the capability to reach the fixpoint, by entering in (the strong basin of) the cyclic attractor.

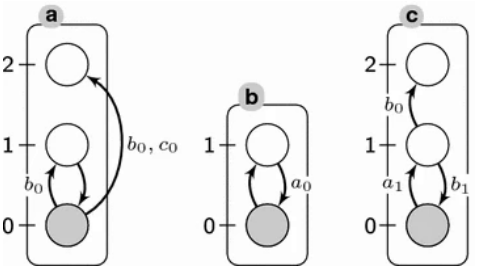

Fig. 1. Running example: Automata network, from [10]; grey-shaded states are initial

In this paper, we address the computation of strong basins that are reachable from a fixed initial state. We rely on concurrency theory to obtain compact representations of reachable sequences of transitions by means of *safe Petri nets* unfoldings, which avoid the explicit exploration of all interleavings. They provide a compact and insightful representation of strong basins, by focusing on the causality and context of transitions.

Safe (or 1-bounded) Petri nets [20] are close to Boolean and multivalued networks [3], although they enable a more fine-grained specification of the conditions for triggering value changes. Focussing on safe PNs entails no limitation of generality of the model, as two-way behaviour-preserving translations between boolean and multivalued models exist (see [3] and the appendix of [2] for discussion). Indeed, instead of a function whose evaluation gives the next value for each node of the network, Petri nets explicitly specify the nodes that actually enable each value change, which are typically very few. *Unfoldings* [8] of Petri nets, which are essentially event structures in the sense of Winskel et al. [21] with additional information about *states* that are crucial in our work here, bring an acyclic representation of the possible sequences of transitions,

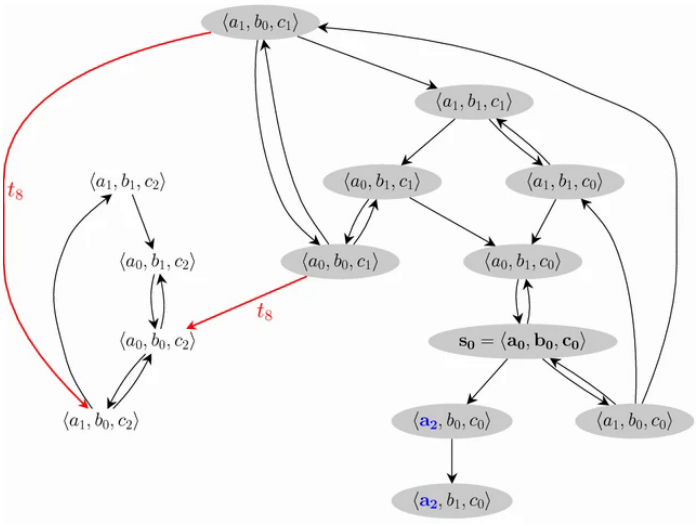

Fig. 2. Transition graph for the automata network of Figure 1, from [10]. Attractors are $\{\langle a_2, b_1, c_0\}$ and $\{\langle a_1, b_1, c_2\rangle, \langle a_0, b_1, c_2\rangle, \langle a_0, b_0, c_2\rangle, \langle a_1, b_0, c_2\rangle\}$.

akin to Mazurkiewicz traces [7][1] but enriched with branching information. From an unfolding, all reachable states and even attractors [2] can be extracted. Like the authors of [11] did for transition systems, we will exploit both forward and reverse dynamics, by constructing reverse unfoldings to explore co-reachable states (i.e. states *from which* a given set can be reached).

Here again, the restriction to 1-safe nets is a technical convenience for unfoldings but not a strict necessity; the use of finite complete prefixes and application of our techniques is possible for general *bounded*[2] nets via careful model translations, not discussed here due to lack of space. Also, there is a variety of other formal approaches that use concurrency to avoid state space explosion, such as partial order reduction in reachability-related tasks following Godefroid [12], or dihomotopy in the sense of Goubault [13]; their angle of attack is on the level of transition systems, whereas our approach focusses on local causal relations.

Outline. After providing the necessary definitions for Petri nets and their unfoldings in Section 2, we will turn our attention to attractors in Section 3. The algorithm that we had developped in [2] for *finding* the complete list of attractors in a safe Petri net, will be extended here. The extended algorithm ATTMAP below provides, in addition to the attractors, information about which system state, called *marking* for Petri nets, allows to reach which attractor. The output of ATTMAP includes a function $\mathsf{Sig}(\bullet)$ that assigns to every marking

[1] which Cousot at al. [6] have recently use as theoretical foundation for capturing which entity of a program is *responsible* of a given behavior

[2] note that infinite-state Petri nets do not have finite complete prefixes in our sense.

the set of attractors in whose basin of attraction the marking lies. Inversely, the strong basin of an attractor $\mathbf{A}$ consists precisely of those markings M for which $\mathsf{Sig}(M) = \{\mathbf{A}\}$. The second, and final, step is then taken in Section 5: we develop two algorithmic methods (on-line and off-line) built on the same principle; they delimit the strong basin of any attractor of the net, via a reverse Petri net unfolding that explores the possible processes that might have led into a given attractor. Truncating this unfolding at the boundary of the strong basin allows to exhibit the strong basin as the set of *interior configurations* in the sense defined below, of the net structure obtained from unfolding. Section 6 concludes.

2 Petri Nets and Unfoldings

Petri Nets. A Petri net is a bipartite directed graph where nodes are either *places* or *transitions*, and places may carry *tokens*. In this paper, we consider only *safe* Petri nets where a place is either active or inactive (as opposed 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 currently active places form the state, or *marking*, of the net. A transition is called enabled in a marking if all its input places are active, and no output place is active unless it is also an input place. The *firing* of a transition modifies the current marking of the net by rendering the input places inactive and output places active.

Formally, a *net* is a tuple $N = (P, T, F)$, where T is a set of *transitions*, P a set of *places*, 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*. A *Petri net* is a tuple $\mathcal{N} = \langle N, M_0 \rangle$, with $M_0 \subseteq P$ an *initial marking*.

In figures, places are represented by circles and the transitions by boxes (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. The *reverse net* of $\mathcal{N}$ is $\overleftarrow{\mathcal{N}} \stackrel{def}{=} (P, T, F^{-1})$. 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, denoted $M \xrightarrow{t}$, if and only (i) ${}^\bullet t \subseteq M$, and (ii) $(M \cap t^\bullet) \subseteq {}^\bullet t$. Note that the second requirement is usually not made in the Petri net literature; however in the class of safe nets (see below), condition (ii) is true whenever (i) is true. An enabled transition t can *fire*, leading to the new marking $M' = (M \setminus {}^\bullet t) \cup t^\bullet$; in that case write $M \xrightarrow{t} M'$. A *firing sequence* is a (finite or infinite) word $w = t_1 t_2 t_3 \ldots$ over T such that there exist markings $M_1, M_2, \ldots$ such that $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} M_2 \xrightarrow{t_3} \ldots$. All markings in such a firing sequence are called *reachable* from the initial marking M_0. We denote the set of markings reachable from some marking M in $\mathcal{N}$ by $\mathbf{R}_{\mathcal{N}}(M)$ (dropping the subscript $\mathcal{N}$ if no confusion can arise).

A marking M reachable in $(\overleftarrow{N}, M_0)$ is said *co-reachable* from M_0 in N; denote as $\overleftarrow{\mathbf{R}}_N(M_0)$ the set of co-reachable markings for M_0 and $\mathcal{N}$.

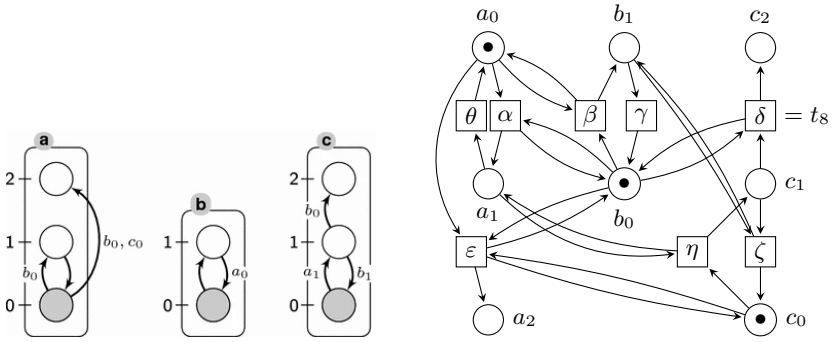

Fig. 3. Right hand side: A Petri net version of the automata network from Figure 1, reproduced on the left hand side.

Semantics of Safe Petri Nets. As an important restriction, we require all Petri nets arising in this paper to be *safe*, that is, in any reachable marking M and any transition t such that ${}^\bullet t \subseteq M$, one has $(M \cap t^\bullet) \subseteq {}^\bullet t$. For an easier formalization of the following concepts, it is possible to introduce *complementary places*: a place $\overline{p}$ is a complement of place p iff

1. ${}^\bullet p = \overline{p}^\bullet$ and $p^\bullet = {}^\bullet\overline{p}$, and
2. $[M_0 \cap \{p, \overline{p}\}| = 1$.

A Petri net $\mathcal{N} = (P, T, F, M_0)$ is called *complete* iff every place in P has at least one complement in P. Complete Petri nets are safe by construction, since the number of tokens on a pair of complementary places is an invariant of the transition firing. Moreover, the reverse net of a complete net is also complete, and thus safe. For a net $N = (P, T, F)$, the *completion* of N is $\hat{N} = (P \uplus \overline{P}, T, F \uplus \overline{F})$, where $\overline{P} = \{\, \overline{p} \mid p \in P \,\}$ is disjoint from P, and

$$\overline{F} \stackrel{def}{=} \{\, (\overline{p}, t) \mid (t, p) \in F \cap (T \times P) \,\} \cup \{\, (t, \overline{p}) \mid (p, t) \in F \cap (P \times T) \,\}$$

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 some initial marking of the net are the terminal strongly connected components of the associated reachability graph.

Definition 1. *Let $N = (P, T, F)$ be a net and $\mathcal{M}$ a set of markings. The* marking graph *induced by $\mathcal{M}$ is a directed graph $(\mathcal{M}, \mathcal{E})$ such that $\mathcal{E} \subseteq \mathcal{M} \times \mathcal{M}$ contains (M, M') if and only if $M \xrightarrow{t} M'$ for some $t \in T$; the arc (M, M') is then* labeled *by t. The* reachability graph *of a Petri net $(\mathcal{N}, M_0)$ is the marking graph $\mathcal{G}(N, M_0)$ induced by $\mathbf{R}_\mathcal{N}(M_0)$. The* transition system *of a complete net $N = (P \uplus \overline{P}, T, F \uplus \overline{F})$ is the marking graph $\mathcal{TS}(N)$ induced by $\{\, M \cup \overline{P \setminus M} \mid M \subseteq 2^P \,\}$.*

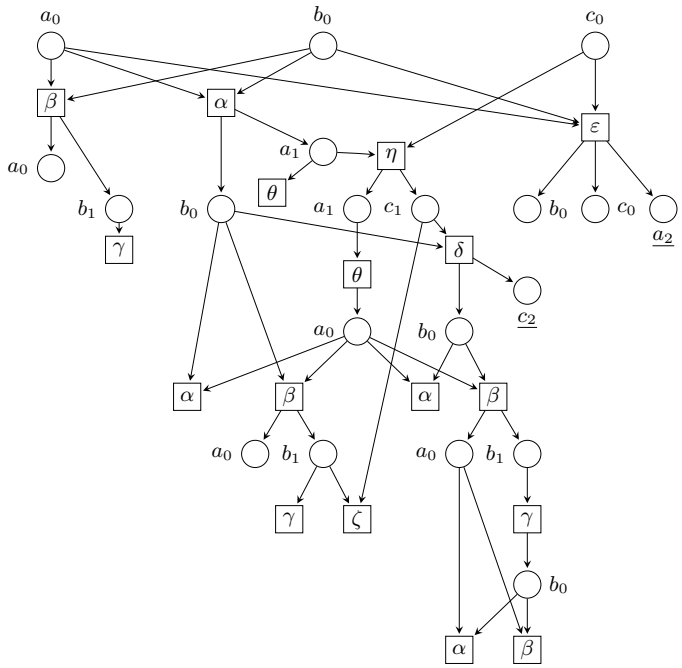

Fig. 4. A finite complete prefix of the unfolding of the Petri net of Figure 3; underlined names indicate fixed conditions, events without outgoing arcs are cut-offs.

Note that complementary places are uniquely defined. To keep the presentation legible and compact, we will henceforth drop all complement places from both notations and figures.

Unfoldings. Let us now recall the basics of Petri net unfoldings and how to use them in finding attractors, following [2].; . Roughly speaking (a more extensive treatment can be found, e.g., in [8]), 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 to compute a finite prefix $\preceq$ of $\mathcal{U}$ that is "complete" in the sense that every reachable marking of $\mathcal{N}$ has a reachable counterpart in $\preceq$, and vice versa [18,8].

Definition 2 (Causality, conflict, concurrency). *Let $N = \langle P, T, F\rangle$ be a net and $x, y \in P \cup T$ two nodes of 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 $u \neq v$, $u \leq x$, $v \leq y$, and ${}^\bullet u \cap {}^\bullet v \neq \emptyset$. We call x and y* concurrent*, noted x co y, if they are neither causally related nor in conflict.*

Definition 3 (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:*

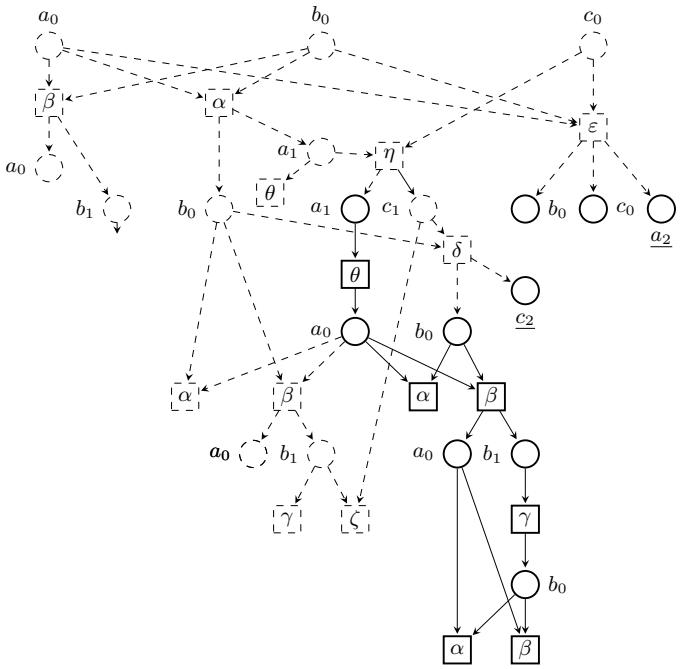

Fig. 5. The attractors for the Petri net of Figure 3, represented in the unfolding prefix of Figure 4.

1. *The causality relation* $<$ *is acyclic;*
2. $|{}^\bullet p| \leq 1$ *for all places* p*, and* $p \in M_0$ *iff* $|{}^\bullet p| = 0$*;*
3. *For every transition* t*,* $t \# t$ *does not hold, and* $[x] \stackrel{def}{=} \{x \mid x \leq t\}$ *is finite.*

We say that $\mathcal{N}$ *is a* reverse occurrence net *iff* $\overleftarrow{\mathcal{N}}$ *is an occurrence net.*

As we said before, an *unfolding* is an "acyclic" version of a safe Petri net $\mathcal{N}$. This notion of acyclicity is captured by Definition 3.

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 4 (Configuration, cut). *Let* $\mathcal{N} = \langle B, E, G, \mathbf{c}_0 \rangle$ *be an occurrence net. A set* $C \subseteq E$ *is called* configuration *of* $\mathcal{N}$ *if (i)* C *is* causally closed*, i.e.* $e' < e$ *and* $e \in E$ *imply* $e' \in E$*; and (ii)* C *is conflict-free, i.e. if* $e, e' \in C$*, then* $\neg(e \# e')$*. The* cut *of* C*, denoted* $\mathbf{cut}(C)$*, is the set of conditions* $(\mathbf{c}_0 \cup C^\bullet) \setminus {}^\bullet C$*.*

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. Note that $\emptyset$ is a configuration, and that $\mathbf{c}_0$ is its cut.

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 $\mathcal{U} = \langle B, E, G, \mathbf{c}_0 \rangle$ 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 condition set B is a subset of $(E \cup \{\bot\}) \times P$. For a condition $b = \langle e, p \rangle$, we will have $e = \bot$ iff $b \in \mathbf{c}_0$; otherwise e is the singleton event in ${}^\bullet b$. Moreover, $h(b) = p$. The initial marking $\mathbf{c}_0$ contains exactly one condition $\langle \bot, p \rangle$ for each initially marked place $p \in M_0$ of $\mathcal{N}$.
2. The events of E are a subset of $2^B \times T$. More precisely, for every cut $\mathbf{c}$ and $B' \subseteq \mathbf{c}$ such that $\{\, h(b) \mid b \in B' \,\} = {}^\bullet t$, we have an event $e = \langle B', t \rangle$. In this case, we add edges $\langle b, e \rangle$ for each $b \in B'$ (i.e. ${}^\bullet e = B'$), we set $h(e) = t$, and for each $p \in t^\bullet$, we add to B a condition $b = \langle e, p \rangle$ connected by an edge $\langle e, b \rangle$.

Intuitively, a condition $\langle e, p \rangle$ represents the possibility of putting a token onto place p through a particular firing sequence, while an event $\langle B', e \rangle$ represents a possibility of firing transition e in a particular context.

Recall that a finite configuration 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 $\mathit{Mark}(C) := \{\, h(b) \mid b \in \mathbf{cut}(C) \,\}$. 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 5 (complete prefix). *Let $\mathcal{N} = \langle P, T, F, M_0 \rangle$ be a safe Petri net and $\mathcal{U} = \langle B, E, G, \mathbf{c}_0 \rangle$ its unfolding. A finite occurrence net $\preceq = \langle B', E', G', \mathbf{c}_0 \rangle$ is said to be a* prefix *of $\mathcal{U}$ if $E' \subseteq E$ is causally closed, $B' = \mathbf{c}_0 \cup E'^\bullet$, and G' is the restriction of G to B' and E'. A prefix $\preceq$ is said to be* complete *if for every reachable marking M of $\mathcal{N}$ there exists a configuration C of $\preceq$ such that (i) $\mathit{Mark}(C) = M$, and (ii) for each transition $t \in T$ enabled in M, there is an event $\langle B'', t \rangle \in E'$ enabled in $\mathbf{cut}(C)$.*

We shall write $\Pi_0(\mathcal{N}, M)$ to denote an arbitrary complete prefix of $\mathcal{N}$ from initial marking M. It is known [18,9] that the construction of such a complete prefix is indeed possible, and efficient tools [22,14] 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". The construction then does not continue beyond any such cut-off event.

3 Attractors

Definitions and Fundamental Properties. The notion of *attractor* denotes, informally, a set of states from which the system cannot 'escape', i.e. from any

state of an attractor, only states inside the same attractor are reachable; that is, the attractors are exactly the terminal SCCs of the transition system. The *strong basin* of an attractor $\mathbf{A}$ collects those states from which the system eventually enters $\mathbf{A}$ (never to leave it again), and its *weak basin* those states from which the system may enter $\mathbf{A}$.

Definition 6. *Let $\mathcal{N} = (P, T, F)$ be a net. An* attractor $\mathbf{A} \subseteq 2^P$ *is a bottom (terminal) strongly connected component (SCC) of $\mathcal{TS}(\mathcal{N})$. Denote the set of attractors (of $\mathcal{N}$) by $\mathcal{A}$, and the set of attractors reachable from a marking M by*

$$\mathsf{Sig}(M) \stackrel{def}{=} \{\mathbf{A} \in \mathcal{A} : \ \mathbf{A} \cap \mathbf{R}(M) \neq \emptyset\}.$$

In particular, an attractor $\mathbf{A}$ is a fixed point *iff there is $M \in \mathcal{M}$ such that $\mathbf{A} = \{M\}$, and for any $t \in T$, $M \stackrel{t}{\rightarrow} M'$ implies $M = M'$.*

It is important to stress the fact that the SCCs to be considered as attractors have to be terminal. In the example, the system may (though this is not likely) cycle forever in the set of states in which neither a_2 nor c_2 holds; however, this set is not an attractor. When we wish to give a dynamic characterization of attraction, and in particular of *basins of attraction*, it is not enough to require the existence of infinite runs that stay inside a given state set; we need to restrict to those runs that 'eventually explore all accessible branches'. This intuition can be captured by the notion of *fairness*: any transition that is *enabled* infinitely often, must also eventually occur :

Definition 7. *In $\mathcal{N}$ as above, an infinite firing sequence $M_0 \stackrel{t_1}{\rightarrow} M_1 \stackrel{t_2}{\rightarrow} \ldots$ is* fair *iff for all $t \in T$:*

$$\left|\left\{i \in \mathbb{N};\ M_i \stackrel{t}{\rightarrow}\right\}\right| = \infty \Rightarrow \{j \in \mathbb{N};\ t_j = t\} \neq \emptyset \tag{1}$$

Note that this notion corresponds to *weak* fairness in the sense of [23], which is sufficient for our purposes (see also Abadi et al [1]); we thus speak only of fairness here. Any such fair sequence will eventually leave any spurious SCC, and, the state space of the net is finite, sooner or later enter a bottom SCC, which, of course, it cannot leave anymore. We are thus ready to define:

Definition 8. *Let $\mathcal{A}' \subseteq \mathcal{A}$. The* strong basin $\mathcal{B}_{\mathcal{A}'} \subseteq 2^P$ *of $\mathcal{A}'$ is the set of markings from which every fair firing sequence leads eventually into some $\mathbf{A} \in \mathcal{A}'$; the* weak basin $\mathcal{W}_{\mathcal{A}'} \subseteq 2^P$ *of $\mathcal{A}'$ is the set of markings from which some $\mathbf{A} \in \mathcal{A}'$ is reachable. By abuse of notation, we will write $\mathcal{B}_{\mathbf{A}}$ ($\mathcal{W}_{\mathbf{A}}$) for $\mathcal{B}_{\mathcal{A}'}$ ($\mathcal{W}_{\mathcal{A}'}$) when $\mathcal{A}' = \{\mathbf{A}\}$.*

Note that for all attractors $\mathbf{A}$, we have $\mathbf{A} \subseteq \mathcal{B}_{\mathbf{A}} \subseteq \mathcal{W}_{\mathbf{A}}$. Also, two distinct attractors $\mathbf{A}, \mathbf{A}'$ must be disjoint, and their *strong* basins too. However, two *weak* basins $\mathcal{B}_{\mathbf{A}}, \mathcal{B}_{\mathbf{A}'}$ are *never* disjoint, as each contains at least M_0.

Signatures for Configurations. Lifting the notion of attraction to the level of *configurations*, and by abuse of notation, we set, for any configuration C, Hence,

$$\mathsf{Sig}(C) \stackrel{def}{=} \mathsf{Sig}(\mathit{Mark}(C)).$$

Note that in general, for any M there will be several C such that $\mathit{Mark}(C) = M$.

4 Extracting attractors from unfoldings

In this section, we present a new method that identifies, for a given Petri net $\mathcal{N}$, both its attractors and their basins, based on the unfolding of $\mathcal{N}$. We first recall the method from our previous work [2] that identifies attractors, and then present the new algorithm.

Representation of attractors as finite complete prefixes. The method from [2] uses unfoldings in two ways: first to find a set of markings which intersects all the attractors, and secondly to output the attractors as a set of finite complete prefixes.

Every attractor $\mathbf{A}$ can be compactly represented as a finite complete prefix of the unfolding of the Petri net $\mathcal{N}$ initialized at some marking $M \in \mathbf{A}$. Let us 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. Lastly, 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.

Maximal configurations and attractors. Let us recall from [2]:

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

The Attractor Map. The following algorithm generates a 'map' of attractors, i.e. the set of these attractors together with the information which marking obtained from a maximal configuration of the first complete prefix leads into which attractor. It is an extension of the algorithm from [2] for *finding* attractors.

Algorithm AttMap. **Initialize** $\hat{\rightsquigarrow} \stackrel{def}{=} \emptyset$, $\mathcal{A}^* \stackrel{def}{=} \emptyset$ and $\hat{\mathcal{A}} \stackrel{def}{=} \emptyset$.

1. Compute a finite complete prefix Π_0 of the unfolding of $\mathcal{N}$; initialise $\Pi \stackrel{def}{=} \Pi_0$.
2. Initialize $\mathcal{M}$ to the set $\mathcal{M}_{\max}$ of markings corresponding to maximal configurations of Π_0.
3. Loop: for M in $\mathcal{M}$ do
   - Compute a finite complete prefix Π_M of the Petri net $\mathcal{N} = (N, M)$. Grow Π by appending a copy of Π_M to every configuration C_M of Π such that $\mathit{Mark}(C_M) = M$.

- Compute the set $\mathbf{next}_M \stackrel{def}{=} \{M' \in \mathcal{M}\backslash\{M\} : \ M' \in \mathbf{R}(M)\}$ of markings in $\mathcal{M}$ that are reachable from M (reachability check done using Π_M).
  - If $\mathbf{next}_M \neq \emptyset$ then update $\hat{\rightsquigarrow} := \hat{\rightsquigarrow} \cup (\{M\} \times \mathbf{next}_M)$;
  - If $\mathbf{next}_M = \emptyset$ then update $\hat{\mathcal{A}} := \hat{\mathcal{A}} \cup \{\Pi_M\}$ and $\mathcal{A}^* := \mathcal{A}^* \cup \{M\}$.

4. Output the attractor candidates, i.e. marking set $\mathcal{A}^*$ and the set $\hat{\mathcal{A}}$ of unfolding prefixes, and the transitive closure $\rightsquigarrow \stackrel{def}{=} (\hat{\rightsquigarrow})^*$ of $\hat{\rightsquigarrow}$.
5. Define the equivalence relation $\equiv$ on $\mathcal{A}^*$ by $M \equiv M' \stackrel{\triangle}{\Leftrightarrow} M \rightsquigarrow M' \ \wedge \ M' \rightsquigarrow M$
6. In the quotient of $\mathcal{A}^*$ under $\equiv$, one obtains the set of root markings of attractors as the set $\underline{\mathcal{A}^*}$ of $\rightsquigarrow$-maximal elements; the set $\mathcal{A}$ of attractors is the set of prefixes rooted in some marking from $\underline{\mathcal{A}^*}$.
7. Compute $\mathsf{Sig}(\bullet)$:
   - For every marking $M \in \underline{\mathcal{A}^*}$, $\mathsf{Sig}(M) \stackrel{def}{=} [M]_\equiv$.
   - For other $M \in \mathcal{M}$: $\mathsf{Sig}(M) \stackrel{def}{=} \{[M']_\equiv : \ M' \in \hat{\mathcal{A}} \wedge M \rightsquigarrow M'\}$.
   - For all families of configurations $(C_i)_{i\in\mathcal{I}}$,
     - $\mathsf{Sig}(\bigcap_{i\in\mathcal{I}} C_i) = \bigcup_{i\in\mathcal{I}} \mathsf{Sig}(C_i)$, and
     - if $C_\mathcal{I} \stackrel{def}{=} \bigcup_{i\in\mathcal{I}} C_i \in \mathcal{C}$, then $\mathsf{Sig}(C_\mathcal{I}) = \bigcap_{i\in\mathcal{I}} \mathsf{Sig}(C_i)$.

Note that every attractor can be represented as an occurrence net $\mathbf{A}_M \stackrel{def}{=} \mathcal{U}_M$ rooted at some attractor marking M, and that no M can be the root of (or even belong to the marking set of) two distinct attractors. Moreover, the sets $\mathcal{C}(M)$ contain full information about which marking from $\mathcal{M}$ allows to reach which attractors, via the relation $\rightsquigarrow \subseteq \mathcal{M}^2$.

Comparison with the original algorithm from [2].

- AttMap explores, for every $M \in \mathcal{M}$, *all* markings that are reachable from M, whereas in [2] it was sufficient to detect *existence* of some such markings; the worst-case complexity is thus increased by one exponential factor.
- The information about how some attractor was reached is stored during the procedure, which induces only a bounded increase of computational and storage effort.
- A more fundamental difference is that attractors come out of the [2] algorithm as individual markings, rather than equivalence classes as in AttMap. The reason is that the [2] algorithm discards every marking from which an attractor representative marking is reachable; this reduction is not available in the above since all reachability information between the candidate markings is to be stored. In general, this will include mutual reachabilities; the quotient with respect to $\equiv$ contracts these strongly connected components. Since, by a classical result from order theory, the preorder $\rightsquigarrow$ is collapsed into a partial order by the quotient operation, one retrieves indeed all attractors from the maximal nodes of $\rightsquigarrow_{/\equiv}$.

5 Basins And Their Boundaries

In this section, we will present methods for computing an unfolding-based representation of the strong basin of a given attractor. The two methods represent different tradeoffs w.r.t. time vs space requirements. We will present a so-called *on-line* method first and the *off-line* method later.

On-line Method for Computing Strong Flow Basins. Let N be a net and $\mathbf{A}$ an attractor of N, i.e. a terminal SCC of $\mathcal{TS}(N)$. The following procedure gives a method to compute $\mathcal{B}(\mathbf{A})$.

1. Fix any marking $M \in \mathbf{A}$, and compute $\Phi(M) \stackrel{def}{=} \Pi_0(\overleftarrow{N}, M) = (B, E, F, \mathbf{c}_0)$. Clearly, all $M' \in \mathbf{A}$, and even all $M'' \in \mathcal{W}(\mathbf{A})$, are represented in $\Phi(M)$.
2. Let $\mathcal{M}^{\leftarrow}$ be the set of markings of maximal configurations of $\Phi(M)$.
3. Set $\mathcal{C}_{\mathbf{A}} := \emptyset$. For all $M' \in \mathcal{M}^{\leftarrow}$, do the following:
   (a) Apply the algorithm ATTMAP on the net $(\mathcal{N}, M')$; this computes, among other things, a complete prefix $\Pi(M')$ as well as $\mathsf{Sig}(M'')$ for all markings M'' reachable from M'.
   (b) For all minimal configurations C' of $\Pi(M')$ satisfying $\mathsf{Sig}(C') = \{\mathbf{A}\}$, find a configuration C of $\Phi(M)$ such that $Mark(C) = Mark(C')$, and add C to $\mathcal{C}_{\mathbf{A}}$.
4. With $E' := \bigcup_{C \in \mathcal{C}_{\mathbf{A}}} C$ and $B' = {}^\bullet E' \cup E'^\bullet \cup \mathbf{c}_0$, let $\Psi(\mathbf{A}) = (B', E', F, \mathbf{c}_0)$, i.e. $\Psi(\mathbf{A})$ is the restriction of $\Phi(M)$ to the events of configurations in $\mathcal{C}_{\mathbf{A}}$.

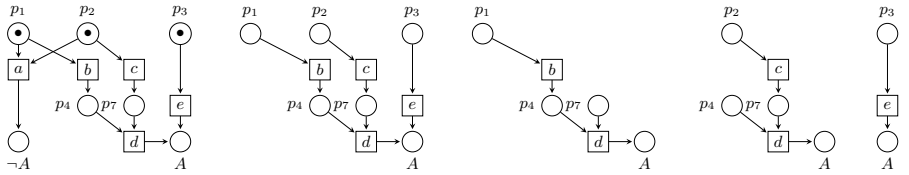

Fig. 6. Illustration of the techniques and concepts used in the computation of $\Psi(\mathbf{A})$. The attractors of the Petri net on the left hand side are the fixed points, i.e. singleton markings, $\mathbf{A} = \{\{A\}\}$ and $\overline{\mathbf{A}} = \{\{\neg A\}\}$. From the initial marking $M_0 = \{p_1, p_2\}$ both attractors are reachable. Reverse exploration from $\mathbf{A}$ yields $\mathcal{M}^{\leftarrow} = \{\{p_1, p_2\}, \{p_3\}\}$. Computing $\Phi(\{A\})$ yields the reverse occurrence net second-from-left. The three interior configurations of $\Psi(\mathbf{A})$ are shown in the figures from center to right; the strong basin of $\mathbf{A}$ is the collection of these configurations and their suffixes.

$\mathcal{B}(\mathbf{A})$ is now represented by $\Psi(\mathbf{A})$ and $\mathcal{C}_{\mathbf{A}}$ in the sense that $M_1 \in \mathcal{B}(\mathbf{A})$ iff there exists a configuration $C' \subseteq C$ with $Mark(C') = M_1$ and $C \in \mathcal{C}$. We shall call $\mathcal{C}_{\mathbf{A}}$ the *interior configurations* of $\mathbf{A}$, motivated by the following two results:

Lemma 1. *Let $C' \subseteq C$ be a configuration of $\Psi(\mathbf{A})$ such that $C \in \mathcal{C}_{\mathbf{A}}$. Then $Mark(C') \in \mathcal{B}(\mathbf{A})$.*

Proof: Let $M_1 = Mark(C')$ and $M'' = Mark(C)$. Since $C' \subseteq C$, we have that M'' is reachable from M_1 in $\overleftarrow{N}$, and hence M_1 is reachable from M'' in N. Since $\mathsf{Sig}(M'') = \{\mathbf{A}\}$, the only attractor reachable from M_1 can be $\mathbf{A}$, too. Moreover, $\mathbf{A}$ is indeed reachable from M_1 because C' is a configuration of $\Phi(M)$, and therefore reachable in $\overleftarrow{N}$ from M, where $M \in \mathbf{A}$. □

Lemma 2. *For every state $M_1 \in \mathcal{B}(\mathbf{A})$, there is a configuration C' of $\psi(\mathbf{A})$ such that $Mark(C') = M_1$ and $C' \subseteq C$ for some $C \in \mathcal{C}_{\mathbf{A}}$.*

Proof: In step 1 of the algorithm, $\Phi(M)$ represents all markings, including M_1, that can reach $\mathbf{A}$ in N and hence M. Therefore, in $\Phi(M)$ there exists some configuration whose marking is M_1 and which reaches some maximal configuration of $\Phi(M)$. Thus, among the maximal markings in $\mathcal{M}^{\leftarrow}$, there must be some M' such that M' reaches M through M_1. On each such path there must be a first marking M'' with $\mathsf{Sig}(M'') = \{\mathbf{A}\}$, i.e. M'' can reach only $\mathbf{A}$ and no other attractor. Since $\Pi(M')$ contains all markings reachable from M', step 3(b) of the algorithm is bound to find some configuration C' whose marking is such an M''. Since M'' is also reachable from M in $\overleftarrow{N}$, it is represented in $\Psi(M)$, and therefore a configuration C with $Mark(C) = M''$ will be added to $\mathcal{C}_{\mathbf{A}}$. □

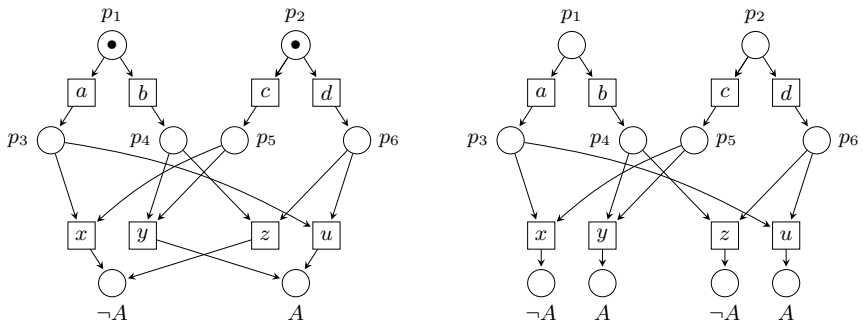

Fig. 7. A net (left) and its complete unfolding (right) with a bifurcation between two attractors, $\{A\}$ and $\{\neg A\}$ that requires coordination of two choices. From markings $\{p_3, p_5\}$ and $\{p_4, p_6\}$, it is inevitable to reach $\{\neg A\}$. That is, the two concurrent choices - between a and b on the one hand, and c and d on the other - must be coordinated to ensure that, e.g., $\{A\}$ is eventually reached; no local choice can achieve this alone.

Note that in general the set of configurations of net $\Psi(\mathbf{A})$ overapproximates the basin, as there may be non-interior configurations spanned by $\Psi(\mathbf{A})$. The example in Figure 6 illustrates this point, see the discussion in the caption.

Off-line computation. In step 3 of the above on-line method, the algorithm ATTMAP is called at runtime, every time some marking M' from $\mathcal{M}^{\leftarrow}$ is inspected. An alternative procedure, which we call *off-line*, would consist in computing, before any basin is inspected, the signature for all states of the transition system of N; then, the value of $\mathsf{Sig}(C)$ would be found, when needed, by a lookup. Indeed, this computation can be implemented by applying ATTMAP to a modified complete net with places $P \cup \overline{P}$: add a new place p_0 to P and its complement place $\overline{p_0}$ to $\overline{P}$. Make $M'_0 := P \cup \{p_0\}$ the initial marking, and for each place $p \in P$, add a transition t_p with ${}^\bullet t_p = \{p_0, p\}$ and $t_p{}^\bullet = \{p_0, \overline{p}\}$.

Add a further transition t_0 from p_0 to $\overline{p_0}$, i.e. ${}^\bullet t_0 = \{p_0\}$ and $t_0{}^\bullet = \{\overline{p_0}\}$, and add $\overline{p_0}$ to the presets and postsets of all the original transitions of N. In this way, the modified net can first decide to move to any marking $M \subseteq P$ before firing t_0 and then behaving just like $\mathcal{N}$ would, if starting at M. The advantage of the off-line method is that it avoids computing the same signatures several times, in the case of overlaps, and that it produces a very rich set of information about the dynamics of a net, for any further use. Moreover, the computation of $\Phi(M)$ can be limited and treat events e with $Mark([e]) \neq \{\mathbf{A}\}$ as cut-off points. Its drawback is in the potentially very big data structures to be explored; the on-line method may be preferred if the system exhibits many small basins, limiting the number of signatures that are actually required. Which method is to be preferred, will need to be decided for every net individually.

Example. It is worth noting that the entry into a strong basin need *not* be linked to a unique transition; depending on the context, the same transition may lead either towards or away from some attractor. Consider Figure 7, letting $M_1 \stackrel{def}{=} \{p_3, p_5\}$ and $M_2 \stackrel{def}{=} \{p_4, p_6\}$. Indeed, the predecessors of M_1 are $\{p_1, p_5\}$ and $\{p_3, p_2\}$, both of which are in the weak basins of both A and $\neg A$, and thus not in $\mathcal{B}_{\neg A}$ (the case for M_2 is symmetric). That is, any transition from $\{a, b, c, d\}$

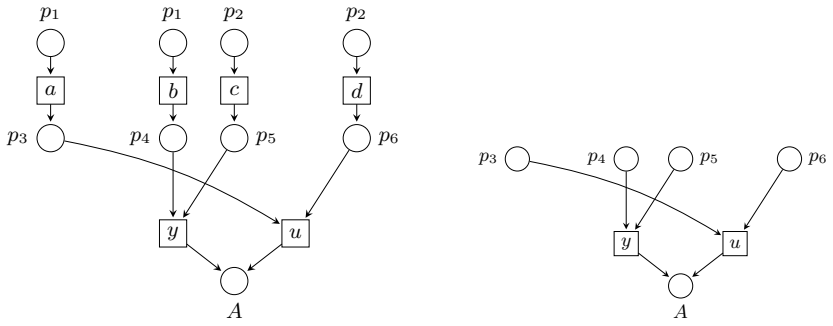

Fig. 8. Continuation of the example from Figure 7. Left: backward unfolding from A to obtain $\Phi(\{A\})$; one has $\mathcal{M}^{\leftarrow} = \{\{p_1\}, \{p_2\}\}$. Right: $\Psi(\{A\})$, allowing two maximal and interior configurations.

may contribute to a path into A, or into $\neg A$; it is the *coordination* among these transitions that decides between the two attractors: occurrence of $\{a, c\}$ or $\{b, d\}$ leads to $\neg A$, that of $\{a, d\}$ or $\{b, c\}$ to A.

Boundaries. We note that the above two cases exhibit two very different behaviours at the boundaries of their strong basins. Let us define a *boundary configuration* to be a configuration C that is not interior, but such that there exists an 'immediate predecessor' interior configuration $C' \subseteq$ with $C \backslash C'$ consisting in a single event. Then, in the example of Figure 6, we have a unique boundary configuration $C = \{b, c, d\}$ with immediate predecessors $C_1 = \{b, d\}$

and $C_2 = \{c, d\}$, and we observe that C is obtained by a sort of closure operation from the interior configurations, in the sense that $C = C_1 \cup C_2$. By contrast, the example of Figures 7 and 8, the boundary configurations are $C^1 = \{a, u\}$, $C^2 = \{d, u\}$, $C^3 = \{b, y\}$, and $C^4 = \{c, y\}$, and the immediate interior successors $C_1 = y$ and $C_2 = \{u\}$. No obvious combination of C_1 and C_2 can produce any C^i. Further classification and study of these (and potentially other) boundary types is left to future work.

6 Conclusion

We have developped Petri net-represented structures that allow to identify completely the strong basins of attraction for all attractors present in a finite safe Petri net. Future work will investigate further the different types of boundaries encountered here, and aim at refining and evaluating *robustness* of attractors and *reprogramming strategies* [17,16] in the context of concurrency. Finally, in regard to benchmarks in prior work relying on Petri net unfoldings [2,4], the time and space consumption of the proposed algorithms allows to envisage their application to networks with two-digit numbers of nodes. In future work, we will investigate the implementation of the on-line and off-line algorithms and their tractability on real-world models of biological systems.

Acknowledgments This research was supported by Agence Nationale de la Recherche (ANR) with the ANR-FNR project AlgoReCell (ANR-16-CE12-0034); Labex DigiCosme (project ANR-11-LABEX-0045-DIGICOSME) operated by ANR as part of the program "Investissement d'Avenir" Idex Paris-Saclay (ANR-11-IDEX-0003-02).

References

1. Martin Abadi and Leslie Lamport. The existence of refinement mappings. *Theor. Comput. Sci.*, 82(2):253–284, 1991.
2. Thomas Chatain, Stefan Haar, Loïg Jezequel, Loïc Paulevé, and Stefan Schwoon. Characterization of reachable attractors using Petri net unfoldings. In Pedro Mendes, editor, *Proceedings of the 12th Conference on Computational Methods in System Biology (CMSB'14)*, volume 8859 of *Lecture Notes in Bioinformatics*, pages 129–142, Manchester, UK, November 2014. Springer-Verlag.
3. Thomas Chatain, Stefan Haar, Juraj Kolcák, Loïc Paulevé, and Aalok Thakkar. Concurrency in Boolean networks. *Natural Computing*, 2019. To appear.
4. Thomas Chatain and Loïc Paulevé. Goal-driven unfolding of petri nets. In Roland Meyer and Uwe Nestmann, editors, *28th International Conference on Concurrency Theory, CONCUR 2017, September 5-8, 2017, Berlin, Germany*, volume 85 of *LIPIcs*, pages 18:1–18:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.
5. David P. A. Cohen, Loredana Martignetti, Sylvie Robine, Emmanuel Barillot, Andrei Zinovyev, and Laurence Calzone. Mathematical modelling of molecular pathways enabling tumour cell invasion and migration. *PLoS Comput Biol*, 11(11):e1004571, 2015.

6. Chaoqiang Deng and Patrick Cousot. Responsibility analysis by abstract interpretation. In Bor-Yuh Evan Chang, editor, *Static Analysis - 26th International Symposium, SAS 2019, Porto, Portugal, October 8-11, 2019, Proceedings*, volume 11822 of *Lecture Notes in Computer Science*, pages 368–388. Springer, 2019.
7. Volker Diekert and Grzegorz Rozenberg, editors. *The Book of Traces*. World Scientific, 1995.
8. Javier Esparza and Keijo Heljanko. *Unfoldings – A Partial-Order Approach to Model Checking*. Springer, 2008.
9. Javier Esparza, Stefan Römer, and Walter Vogler. An improvement of McMillan's unfolding algorithm. *FMSD*, 20:285–310, 2002.
10. Louis Fippo Fitime, Olivier Roux, Carito Guziolowski, and Loïc Paulevé. Identification of bifurcation transitions in biological regulatory networks using Answer-Set Programming. *Algorithms for Molecular Biology*, 12(1):19, 2017.
11. S. Fueyo, P.T. Monteiro, A. Naldi, J Dorier, É Remy, and C. Chaouiya. Reversed dynamics to uncover basins of attraction of asynchronous logical models. *F1000Research*, 30(6), August 2017.
12. Patrice Godefroid. *Partial-Order Methods for the Verification of Concurrent Systems - An Approach to the State-Explosion Problem*, volume 1032 of *Lecture Notes in Computer Science*. Springer, 1996.
13. Eric Goubault and Martin Raußen. Dihomotopy as a tool in state space analysis. In Sergio Rajsbaum, editor, *LATIN 2002: Theoretical Informatics, 5th Latin American Symposium, Cancun, Mexico, April 3-6, 2002, Proceedings*, volume 2286 of *Lecture Notes in Computer Science*, pages 16–37. Springer, 2002.
14. V. Khomenko. Punf. http://homepages.cs.ncl.ac.uk/victor.khomenko/tools/punf/.
15. H. Klarner, H. Siebert, S. Nee, and F. Heinitz. Basins of attraction, commitment sets and phenotypes of boolean networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2018.
16. Hugues Mandon, Cui Su, Stefan Haar, Jun Pang, and Loïc Paulevé. Sequential reprogramming of boolean networks made practical. In Luca Bortolussi and Guido Sanguinetti, editors, *Proceedings of the 17th Conference on Computational Methods in System Biology (CMSB'19)*, volume 11773 of *Lecture Notes in Bioinformatics*, pages 3–19, Trieste, Italy, September 2019. Springer-Verlag.
17. Hugues Mandon, Cui Su, Jun Pang, Soumya Paul, Stefan Haar, and Loïc Paulevé. Algorithms for the sequential reprogramming of boolean networks. *IEEE/ACM Transaction on Computational Biology and Bioinformatics*, 2019. To appear.
18. K. L. McMillan. Using unfoldings to avoid the state explosion problem inthe verification of asynchronous circuits. In *CAV*, pages 164–177, 1992.
19. Nuno D. Mendes, Rui Henriques, Elisabeth Remy, Jorge Carneiro, Pedro T. Monteiro, and Claudine Chaouiya. Estimating attractor reachability in asynchronous logical models. *Frontiers in Physiology*, 9, 2018.
20. T. Murata. Petri nets: Properties, analysis and applications. *Proc. of the IEEE*, 77(4):541–580, 1989.
21. Mogens Nielsen, Gordon D. Plotkin, and Glynn Winskel. Petri nets, event structures and domains, part I. *Theor. Comput. Sci.*, 13:85–108, 1981.
22. S. Schwoon. Mole. http://www.lsv.ens-cachan.fr/ schwoon/tools/mole/.
23. Walter Vogler. Fairness and partial order semantics. *Inf. Process. Lett.*, 55(1):33–39, 1995.