Non-deterministic updates of Boolean networks

Loïc Paulevé, Sylvain Sené

Non-deterministic updates of Boolean networks

Loïc Paulevé
Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, F-33400 Talence, France

Sylvain Sené
Université publique, Marseille, France

Abstract

Boolean networks are discrete dynamical systems where each automaton has its own Boolean function for computing its state according to the configuration of the network. The updating mode then determines how the configuration of the network evolves over time. Many of updating modes from the literature, including synchronous and asynchronous modes, can be defined as the composition of elementary deterministic configuration updates, i.e., by functions mapping configurations of the network. Nevertheless, alternative dynamics have been introduced using ad-hoc auxiliary objects, such as that resulting from binary projections of Memory Boolean networks, or that resulting from additional pseudo-states for Most Permissive Boolean networks. One may wonder whether these latter dynamics can still be classified as updating modes of finite Boolean networks, or belong to a different class of dynamical systems. In this paper, we study the extension of updating modes to the composition of non-deterministic updates, i.e., mapping sets of finite configurations. We show that the above dynamics can be expressed in this framework, enabling a better understanding of them as updating modes of Boolean networks. More generally, we argue that non-deterministic updates pave the way to a unifying framework for expressing complex updating modes, some of them enabling transitions that cannot be computed with elementary and non-elementary deterministic updates.

2012 ACM Subject Classification Theory of computation → Models of computation; Theory of computation → Program semantics; Applied computing → Systems biology

Keywords and phrases Natural computing, discrete dynamical systems, semantics

Funding *Loïc Paulevé*: French Agence Nationale pour la Recherche (ANR): ANR-FNR project "AlgoReCell" ANR-16-CE12-0034 and ANR project "BNeDiction" ANR-20-CE45-0001.
Sylvain Sené: his salary as a French state agent affiliated to Université Aix-Marseille, Université Toulon, CNRS, LIS, UMR 7020, Marseille, France, and the ANR project "FANs" ANR-18-CE40-0002.

1 Introduction

Boolean networks (BNs) are formal dynamical systems composed of automata, each of them having a Boolean state. A major difference between BNs and cellular automata (CAs) is that each automaton of a BN follows its own rules for computing its next state depending on the states of the other automata in the network. Consequently, whereas influences between cells in a CA are structured homogeneously according a cellular space, those between automata in a BN are structured according to any directed graph. In this paper, only finite BNs are considered, as it is generally the case in the literature, notably because BNs are mostly viewed as both a real-world computational model and a real-world modeling framework.

The study of BNs led to fundamental results linking the network architecture (structure of influences between automata) to the existence of fixed points and to the number of limit cycles they can exhibit [1, 10, 4]. Notably, it is well known that such limit behaviors may depend on the way automata update their state over time [3, 12, 2, 22]. This emphasizes the importance of what is classically called the updating modes in the analyses of BNs.

BNs are widely employed to model natural systems, with prominent applications in biology. These applications inspired the definition of various updating modes aiming at reflecting constraints related to the quantitative nature of the abstracted system, such as reaction duration and influence thresholds. There is actually no consensus about one updating modes that would be

the most likely, the most representative of the biological reality. As a consequence, the choice of this or that updating mode strongly depends on the problematics, on the nature of the questions addressed. Thus, it remains essential to analyze the impact of a wide range of updating modes with distinct features.

In this paper, we address the formalization of updating modes in the framework of BNs. From a very general perspective, given a BN and one of its configurations, an updating mode specify how to compute the possible next configurations (plural implying non-deterministic systems).

A large majority of updating modes introduced so far can be expressed using deterministic functions mapping the configurations of the network. This leads to *elementary* transitions, as it is the case with synchronous (or parallel) and asynchronous [23] updating modes, which may result in non-deterministic dynamics. These functions may also be composed, as in block-sequential [25] and block-parallel [11] updating modes, generating non-elementary transitions.

These compositions of deterministic *updates*, however, do not cover all the updating modes introduced in the literature. Indeed, updating modes may also make use of parameters that cannot *a priori* and intuitively be directly captured by these deterministic updates. These parameters can represent kinds of delays or threshold effects of state changes. In this paper, we focus on 3 examples of BN dynamics which have been recently introduced and defined using ad-hoc formalizations:

- *Memory Boolean networks* (MBNs) [14, 15] take into account some kind of delay for the decrease of automata. They have been introduced by the means of a deterministic dynamical system with non-binary configurations, whose updates are computed deterministically from the BN and a memory vector, specifying the delay for each automaton.
- *Interval Boolean networks* (IBNs) [7] account for a duration for updating an automaton. The other automata can be updated until the former automaton eventually change of state. They have been defined by an encoding as the fully-asynchronous updating of a BN of dimension $2n$. The dynamics of the original BN are then recovered by projection.
- *Most Permissive Boolean networks* (MPBNs) [24] bring a formal abstraction of trajectories of quantitative models which are compatible with the BN: from an initial configuration, if there is no trajectory where a given automaton is 1 (or 0), then, no quantitative refinement of the model can increase (or decrease) the value of this automaton. MPBNs have been defined by introducing additional states for automata to account for their state change (increasing and decreasing). An automaton in one of these states can be read non-deterministically as 0 or 1.

Overall, the definition of these BN dynamics involve either non-Boolean configurations, projections of higher-dimension BN, or both. Importantly, they suggest that deterministic updates are not expressive enough to capture specific dynamics. This is striking with IBNs and MPBNs which can generate transitions that are neither elementary nor non-elementary transitions, and thus predict trajectories that are impossible with the asynchronous updating mode.

We show that these dynamics can all be expressed using Boolean configurations in a simple generic framework, which extends the deterministic updates to *non-deterministic* updates: functions mapping sets of configurations. In the case of MBNs, the obtained definition from the binary projections of their deterministic discrete dynamics actually help to understand the generated dynamics: the transitions match with a particular subset of elementary transitions, suggesting a simpler parameterization. In the case of IBNs and MPBNs, the transitions extend the elementary and non-elementary transitions by considering some delay for the state changes, and having different interpretation of how to "read" an automaton in the course of state change. The obtained definitions suggest many variants for generating sub-dynamics, similarly to the asynchronous mode which generates all elementary transitions.

Thus, non-deterministic updates offer a unified yet simple framework for defining and understanding BN updating modes with more expressivity than usual deterministic updates. However,

should any set update be considered as a BN updating mode? We propose an argumentation for a *reasonable* updating mode in the last section, where we suggest that the state change should always be justified by the application of a local function. This suggests that the MP updating mode generates the largest set of transitions that fulfill this criterion.

Notations The Boolean domain $\{0,1\}$ is denoted by $\mathbb{B}$; the set $\{1,\cdots,n\}$ is denoted by $[\![n]\!]$. Given a finite domain A with a partial order $\preceq$, and a function h mapping elements of A to A, for any $k \in \mathbb{N}_{>0}$, we write h^k for h iterated k times. Whenever for any $a \in A$, $a \preceq h(a)$, we write h^ω for the iteration of h until reaching a fixed point (in this paper, A is often a power set with $\preceq$ being the subset relation).

2 Boolean networks and dynamics

A *Boolean network* (BN) of dimension n is specified by a function $f : \mathbb{B}^n \to \mathbb{B}^n$ mapping Boolean vectors of dimension n. The components $[\![n]\!]$ of the BN are called *automata*. For each automaton $i \in [\![n]\!]$, $f_i : \mathbb{B}^n \to \mathbb{B}$ is the i-th component of this function, that we call the *local function* of automaton i. The 2^n Boolean vectors of $\mathbb{B}^n$ are called the *configurations* of the BN. In a configuration $x \in \mathbb{B}^n$, x_i is the *state* of automaton i.

Updating modes Given a BN f of dimension n and one of its configurations $x \in \mathbb{B}^n$, an *updating mode* μ characterizes the possible evolutions of x with respect to $f(x)$. The dynamical system (f,μ) defines a binary *transition relation* between configurations of $\mathbb{B}^n$ denoted by $\rightarrow_{(f,\mu)} \subseteq \mathbb{B}^n \times \mathbb{B}^n$. This dynamical system can be represented by a directed graph $\mathcal{D}_{(f,\mu)} = (\mathbb{B}^n, \rightarrow_{(f,\mu)})$. This graph is usually called the *transition graph* of (f,μ). The reflexive and transitive closure of relation $\rightarrow_{(f,\mu)}$, denoted by $\rightarrow^*_{(f,\mu)}$ can be defined as follows: given two configurations $x, y \in \mathbb{B}^n$, $x \rightarrow^*_{(f,\mu)} y$ if and only if $x = y$ or there exists a path from x to y in $\mathcal{D}_{(f,\mu)}$.

A *deterministic* updating mode ensures that, for any BN f of dimension n, each configuration has at most one outgoing transition ($\forall x, y, z \in \mathbb{B}^n$, $x \rightarrow_{(f,\mu)} y$ and $x \rightarrow_{(f,\mu)} z$ only if $y = z$). Otherwise, the updating mode is qualified as *non-deterministic*.

In the following, we consider the BN f to be fixed, and thus, for the sake of simplicity, we omit the subscript f: the transition relation is denoted by $\rightarrow_\mu$ and the transition graph by $\mathcal{D}_\mu$.

Dynamical properties A configuration $x \in \mathbb{B}^n$ is *transient* if there exists a configuration y such that $x \rightarrow^*_\mu y$ and $y \not\rightarrow^*_\mu x$. Configurations that are not transient are called *limit configurations*. Because n is finite, these configurations induce the terminal strongly connected components of $\mathcal{D}_\mu$, called the *limit sets* of (f,μ). If there exists at least one path from a transient configuration to a limit set, this limit set is called an *attractor* of (f,μ) [8, 21]. The *basin of attraction* of an attractor $\mathscr{A}$ of (f,μ), denoted by $\mathscr{B}(\mathscr{A})$, is the sub-graph of $\mathcal{D}_\mu$ induced by the set of transient configurations x such that, for any limit configuration y belonging to $\mathscr{A}$, $x \rightarrow^*_\mu y$. A limit set of cardinal 1, *i.e.* composed of a unique limit configuration x is called a *fixed point* of (f,μ). A limit set of cardinal greater than 1 is called a *limit cycle* of (f,μ).

3 Updating modes with deterministic updates

Elementary transitions

Let us consider a BN f of dimension n and one of its configurations $x \in \mathbb{B}^n$. Whenever x and $f(x)$ differ by more than one component, one may define several ways to update x: either by replacing it

Table 1 Configurations, local functions ($(f_i)_{i\in[\![3]\!]}$) and four updating functions ($\phi_\emptyset$, ϕ_1, $\phi_{\{2,3\}}$, and $\phi_{[\![3]\!]}$) of Boolean network f presented in Example 2.

$x=(x_1,x_2,x_3)$	$f_1(x)$	$f_2(x)$	$f_3(x)$	$\phi_\emptyset(x)$	$\phi_1(x)$	$\phi_{\{2,3\}}(x)$	$\phi_{[\![3]\!]}(x)\equiv f(x)$
(0,0,0)	1	0	1	(0,0,0)	(1,0,0)	(0,0,1)	(1,0,1)
(0,0,1)	0	1	1	(0,0,1)	(0,0,1)	(0,1,1)	(0,1,1)
(0,1,0)	1	0	1	(0,1,0)	(1,1,0)	(0,0,1)	(1,0,1)
(0,1,1)	0	1	1	(0,1,1)	(0,1,1)	(0,1,1)	(0,1,1)
(1,0,0)	1	0	0	(1,0,0)	(1,0,0)	(1,0,0)	(1,0,0)
(1,0,1)	0	0	0	(1,0,1)	(0,0,1)	(1,0,0)	(0,0,0)
(1,1,0)	1	0	0	(1,1,0)	(1,1,0)	(1,0,0)	(1,0,0)
(1,1,1)	0	0	0	(1,1,1)	(0,1,1)	(1,0,0)	(0,0,0)

with $f(x)$, i.e., applying simultaneously the local functions on every automata, or by modifying the state of only a subset of automata. For each set of automata to update, we obtain a deterministic function mapping configurations, that we refer to as an *elementary* deterministic update:

▶ **Definition 1.** *Given a BN f of dimension n and a set of automata $W\subseteq[\![n]\!]$, $\phi_W:\mathbb{B}^n\to\mathbb{B}^n$ is an* elementary deterministic update *with*

$$\forall x\in\mathbb{B}^n, \forall i\in[\![n]\!],\ \phi_W(x)_i=\begin{cases} f_i(x) & \text{if } i\in W,\\ x_i & \text{otherwise.}\end{cases}$$

Whenever referring to singleton sets $\{i\}$ with $i\in[\![n]\!]$, we write ϕ_i instead of $\phi_{\{i\}}$. Notice that $\phi_{[\![n]\!]}=f$.

▶ **Example 2.** Let us consider the BN f of dimension $n=3$ with $f(x)=\begin{pmatrix} f_1(x)=\neg x_3\\ f_2(x)=\neg x_1\wedge x_3\\ f_3(x)=\neg x_1\end{pmatrix}$.

Table 1 shows four distinct updatings on its configurations. The first updating is ineffective and consists in changing nothing. The second updating changes the state of automaton 1 by application of ϕ_1, the third one changes the states of both automata 2 and 3 by application of $\phi_{\{2,3\}}$, and the fourth one changes the state of every automaton by application of $\phi_{[\![3]\!]}$.

We can then define the notion of *elementary transitions* of a BN, that are the transitions obtained by applying any elementary update on a non-empty subset of automata.

▶ **Definition 3.** *Given a BN f, its* elementary *transitions $\to_{\mathsf{e}}\subseteq\mathbb{B}^n\times\mathbb{B}^n$ are such that, for all configurations $x,y\in\mathbb{B}^n$, $x\to_{\mathsf{e}}y$ if and only if there exists a non-empty subset of automata $W\subseteq[\![n]\!]$ with $y=\phi_W(x)$.*

Let us now define some classical deterministic and non-deterministic updating modes from these elementary updates.

Examples of deterministic updating modes

The most direct updating mode is the application of f to the configuration x, resulting in the configuration $f(x)$, or, equivalently, $\phi_{[\![n]\!]}(x)$:

▶ **Definition 4.** *The* synchronous *(or* parallel*) updating mode of a BN f of dimension n generates the transition relation $\to_{\mathsf{p}}\subseteq\mathbb{B}^n\times\mathbb{B}^n$ such that, for all configurations $x,y\in\mathbb{B}^n$, $x\to_{\mathsf{p}}y$ if and only if $y=\phi_{[\![n]\!]}(x)$.*

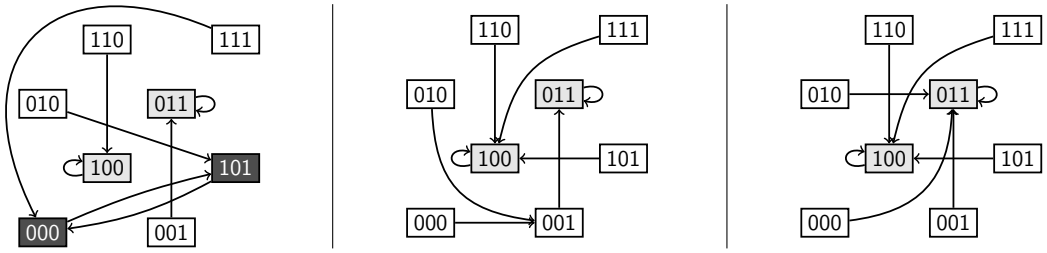

Figure 1 Distinct possible block-sequential dynamics of BN f defined in Example 2: (left panel) its parallel dynamics associated with ordered partition ($[\![3]\!]$); (central panel) the block-sequential dynamics associated with $(\{2,3\},\{1\})$; (right panel) the sequential dynamics associated with $(\{3\},\{1\},\{2\})$.

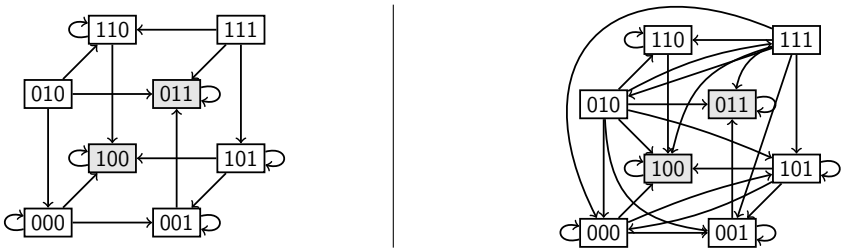

Figure 2 Fully-asynchronous (left) and asynchronous (right) dynamics of BN f defined in Example 2.

Sequential updating modes are parameterized by a permutation of $[\![n]\!]$, fixing an ordering of elementary updates of single automata [13, 17, 9]. They can be generalized to *block-sequential* updating modes [25, 3, 16], parameterized by a permutation of a partition of $[\![n]\!]$:

▶ **Definition 5.** *Given a BN f of dimension n and* $\mathsf{bs} = (W_1, \cdots, W_p)$ *an ordered partition of* $[\![n]\!]$, *the* block-sequential *updating mode generates the transition relation* $\rightarrow_{\mathsf{bs}} \subseteq \mathbb{B}^n \times \mathbb{B}^n$ *such that, for all configurations* $x, y \in \mathbb{B}^n$, $x \rightarrow_{\mathsf{bs}} y$ *if and only if* $y = \phi_{W_p} \circ \cdots \phi_{W_1}(x)$.

Remark that the transitions of sequential and block-sequential modes may not be elementary. However, they always correspond to a path of elementary transitions: $x \rightarrow_{\mathsf{bs}} y$ only if $x \rightarrow_{\mathsf{e}}^* y$.

Going further in generalization, one may consider deterministic updating modes as infinite sequences of sets of automata, so that automata of a same subset execute their local function in parallel while the subsets are iterated sequentially. Remark that any of these possible deterministic updating modes will generate transitions corresponding to specific paths of elementary transitions.

Examples of non-deterministic updating modes

It is important to notice that deterministic updates can lead to non-deterministic dynamics by allowing different updates on a same configuration. The most obvious example is the *asynchronous* mode[1] consisting of all the elementary transitions.

▶ **Definition 6.** *The* asynchronous *updating mode of a BN f generates the transition relation* $\rightarrow_{\mathsf{a}} \subseteq \mathbb{B}^n \times \mathbb{B}^n$ *as* $\rightarrow_{\mathsf{a}} = \rightarrow_{\mathsf{e}}$.

One of the most usual non-deterministic updating modes of BNs is the *fully-asynchronous* mode[2], where only one automaton is updated in a transition. It is largely employed for the analysis

[1] The *asynchronous* mode is often referred to as *general asynchronous* in the systems biology modeling community.
[2] The *fully-asynchronous* mode is usually referred to as *asynchronous* in the system biology modeling community.

of models of biological systems, arguing it enables capturing (some) behaviors caused by different time scale for automata updates.

▶ **Definition 7.** *The* fully-asynchronous *updating mode of a BN f generates the transition relation $\to_{\mathsf{fa}} \subseteq \mathbb{B}^n \times \mathbb{B}^n$ such that, for all configurations $x, y \in \mathbb{B}^n$: $x \to_{\mathsf{fa}} y$ if and only if there exists $i \in [\![n]\!]$ with $y = \phi_i(x)$.*

Figure 2 shows the dynamics generated by the fully-asynchronous and asynchronous updating modes on the BN of Example 2.

4 Non-deterministic updates as set updates

The updates considered so far are deterministic, and can thus be defined as functions mapping configurations, i.e., of the form $\phi : \mathbb{B}^n \to \mathbb{B}^n$. As we have seen above, deterministic updates can generate non-deterministic updating modes, by allowing different updates to be applied on a same configuration.

Let us now extend to non-deterministic updates, that we model by functions mapping *sets of configurations*, i.e., of the form $\Phi : 2^{\mathbb{B}^n} \to 2^{\mathbb{B}^n}$. We define Φ as a map from sets of configurations to sets of configurations for enabling iterations and compositions of non-deterministic updates. Nevertheless, we assume that for any $X \subseteq \mathbb{B}^n$, $\Phi(X) = \bigcup_{x \in X} \Phi(\{x\})$: one can define Φ only from all singleton configuration set. This restriction ensures that, for any $X \subseteq \mathbb{B}^n$, each configuration in the image set $y \in \Phi(X)$ can be computed from a singleton set $\{x\}$ for some $x \in \mathbb{B}^n$. In the following, we call such updates *set updates*.

Starting from a singleton configuration set $\{x\}$, the iteration of set updates delineate the domains of configurations the system can evolve to. Thus, set updates naturally define transition relations between configurations:

▶ **Definition 8.** *Given a set update function Φ for BNs of dimension n, the generated transition relation is given by $\delta : (2^{\mathbb{B}^n} \to 2^{\mathbb{B}^n}) \to 2^{\mathbb{B}^n \times \mathbb{B}^n}$ with $\delta(\Phi) = \{(x, y) \mid x \in \mathbb{B}^n, y \in \Phi(\{x\})\}$.*

In contrast with deterministic updates, non-deterministic updating modes can be characterized directly by set updates. Indeed, non-deterministic updating modes allow "superposing" alternative updates to generate different transitions from a single configuration x, although each of them is computed with a deterministic update. For instance, with one update ϕ where $\phi(x) = y$ and another update ϕ' where $\phi'(x) = y' \neq y$. Now, let us imagine an updating mode superposing two set updates, Φ and Φ' where, for some configurations $x \in \mathbb{B}^n$, $\Phi(\{x\}) \setminus \Phi'(\{x\}) \neq \emptyset$. One can then build a single set update Φ^* such that $\Phi^*(X) = \Phi(X) \cup \Phi'(X)$. It results that $\delta(\Phi^*) = \delta(\Phi) \cup \delta(\Phi')$, thus the updating mode can be assimilated to Φ^*.

Finally, notice that limit sets of the generated dynamics $\delta(\Phi)$ can be characterized as the $\subseteq$-smallest sets of configurations $X \subseteq \mathbb{B}^n$ such that $\Phi(X) = X$.

5 Updating modes selecting elementary transitions

With deterministic updates as building blocks, we have seen that one can define non-deterministic updating modes by superposing different update functions. The resulting transition relation is then the union of the transition relation generated by each individual update (each of them giving a deterministic dynamics). Set updates offer an alternative way to formalize the resulting dynamics, by directly defining the set of out-going transitions from a given configuration. As we will illustrate with the memory updating mode below, this enables a fine-grained selection of the elementary transitions which may then depend on the configuration.

5.1 Asynchronous and fully-asynchronous updating modes

As a first illustration of set updates and how they can characterize updating modes, consider the following set update for BNs of dimension n:

$$\Phi_{\mathsf{e}}(X) = \{\phi_W(x) \mid x \in X, \emptyset \neq W \subseteq [\![n]\!]\}.$$

This set update generates exactly all the elementary transitions: $\delta(\Phi_{\mathsf{e}}) = \rightarrow_{\mathsf{e}}$. Thus, Φ_{e} characterizes the asynchronous updating mode. Similarly, let us now consider the following set update:

$$\Phi_{\mathsf{fa}}(X) = \{\phi_i(x) \mid x \in X, i \in [\![n]\!]\}.$$

Remark that $\delta(\Phi_{\mathsf{fa}}) = \rightarrow_{\mathsf{fa}}$, i.e., Φ_{fa} characterizes the fully-asynchronous updating mode.

5.2 Memory updating mode

Until now, all the updating modes that have been discussed depend on deterministic updates that are context free, which leads to deal with memoryless dynamical systems. In [14, 15] have been introduced another model of BNs, called Memory Boolean networks (MBNs). The first objective of MBNs is to capture the biologically relevant gene-protein BN model introduced in [18], that builds on the following principles:

- automata are split in two types: a half models genes, the other half models their associated one-to-one proteins;
- each protein has its own decay time: the number of time steps during which it remains present in the cell after having been produced by the punctual expression of its associated gene.

In their original definition given below, MBNs of dimension n are BNs of dimension n parameterized with a vector $\mathsf{M} \in \mathbb{N}_{>0}^n$, setting the maximal delay (called memory) for the degradation of each automaton. Then, an automaton is considered active (Boolean 1) whenever its delay to degradation is not 0. Formally, MBN are defined as follows:

▶ **Definition 9.** *A* Memory Boolean network *of dimension n is the couple of a BN f of dimension n and of a memory vector $\mathsf{M} = (\mathsf{M}_1, \ldots, \mathsf{M}_n) \in \mathbb{N}_{>0}^n$. The set of its configurations is defined as $X_{(f,\mathsf{M})} = \{(x,d) \in \mathbb{B}^n \times \mathbb{N}^n \mid \forall i \in [\![n]\!],\ d_i \in \{0,\ldots,\mathsf{M}_i\},\ x_i = 0 \iff d_i = 0 \text{ and } x_i = 1 \iff d_i \in \{1,\ldots,\mathsf{M}_i\}\}$. The dynamical system $((f,\mathsf{M}),\mathsf{p})$ is defined by the transition graph $\mathcal{D}_{((f,\mathsf{M}),\mathsf{p})}$, with p the parallel updating mode, made of transitions based on updating function $\phi^\star : X_{(f,\mathsf{M})} \to X_{(f,\mathsf{M})}$ depending on the memories such that:*

$$\forall (x,d),(y,d') \in X_{(f,\mathsf{M})},\ (x,d) \rightarrow_{((f,\mathsf{M}),\mathsf{p})} (y,d') \iff (y,d') = \phi^\star_{[\![n]\!]}(x,d),$$

where $\forall i \in [\![n]\!]$, $\phi^\star_{[\![n]\!]}(x,d)_i = (y_i, d'_i)$, with:

$$d'_i = \begin{cases} 0 & \text{if } f_i(x) = 0 \text{ and } d_i = 0, \\ d_i - 1 & \text{if } f_i(x) = 0 \text{ and } d_i \geq 1, \\ \mathsf{M}_i & \text{if } f_i(x) = 1, \end{cases} \quad \text{and} \quad y_i = \begin{cases} 1 & \text{if } d'_i \geq 1, \\ f_i(x) & \text{if } d'_i = 0. \end{cases}$$

From this initial definition, it is easy to see that the dynamics of a MBN is deterministic and operates on discrete configurations that are not Boolean anymore. But we will see that MBNs enable to develop a new updating mode, called the memory updating mode, that operates directly on Boolean configurations.

First, let us define $\alpha(x)$ the set of memory configurations corresponding to any binary configuration $x \in \mathbb{B}^n$, and conversely, $\beta(d)$ the binary configuration corresponding to a memory configuration $d \in \mathbb{N}^n$. Notice that $\forall x \in \mathbb{B}^n$, $\forall d \in \alpha(x)$, $\beta(d) = x$.

$$\alpha(x) = \{d \in \mathbb{N}^n \mid x_i = 0 \Leftrightarrow d_i = 0, x_i = 1 \Leftrightarrow d_i \in [\![\mathsf{M}_i]\!]\}$$

$$\forall i \in [\![n]\!] \quad \beta(d)_i = \min\{d_i, 1\}.$$

It appears that $X_{(f,\mathsf{M})} = \{(\beta(d), d) \mid d \in \mathbb{N}^n, \forall i \in [\![n]\!], d_i \in \{0, \dots, M_i\}\}$. Thus one can reformulate the original definition by considering the deterministic parallel update of memory configurations $d \in \mathbb{N}^n$, and replacing x with $\beta(d)$: an automaton $i \in [\![n]\!]$ is set to state M_i whenever its local function f_i is evaluated to 1 on the corresponding binary configuration $\beta(x)$; otherwise, its state is decreased by one, unless it is already 0. In particular, one can define the deterministic memory update $\phi^*_{\mathsf{M}} : \mathbb{N}^n \to \mathbb{N}^n$ such that, for each $i \in [\![n]\!]$,

$$\phi^*_{\mathsf{M}}(d)_i = \begin{cases} 0 & \text{if } f_i(\beta(d)) = 0 \text{ and } d_i = 0, \\ d_i - 1 & \text{if } f_i(\beta(d)) = 0 \text{ and } d_i \geq 0, \\ \mathsf{M}_i & \text{if } f_i(\beta(d)) = 1. \end{cases}$$

Let us now extend the above definitions to sets:

- $\forall X \subseteq \mathbb{B}^n$, $A(X) = \bigcup_{x \in X} \alpha(x)$;
- $\forall D \subseteq \mathbb{N}^n$, $B(D) = \{\beta(d) \mid d \in D\}$;
- $\forall D \subseteq \mathbb{N}^n$, $\Phi^*_{\mathsf{M}}(D) = \{\phi^*_{\mathsf{M}}(d) \mid d \in D\}$.

The memory set update can then be defined for any set of configurations $X \subseteq \mathbb{B}^n$ by first generating the set of corresponding memory configurations, then applying the deterministic update on them, and finally converting them back to binary configurations:

$$\Phi_{\mathsf{M}}(X) = B \circ \Phi^*_{\mathsf{M}} \circ A(X).$$

With this formulation, one can see that the memory updating mode, being the projection of MBN configurations on their binary part, lead to non-deterministic dynamics. Indeed, whenever a configuration gets mapped to several possible memory configurations, and whenever for two of these configurations d and d', there is an automaton $i \in [\![n]\!]$ where $\phi_{\mathsf{M}}(d)_i = 0$ and $\phi_{\mathsf{M}}(d')_i \geq 1$. This can occur if and only if $\mathsf{M}_i \geq 2$, $x_i = 1$, and $f_i(x) = 0$. Thus, the memory updating mode of BNs can equivalently be parameterized by a set of automata $\overline{\mathsf{M}} = \{i \in [\![n]\!] \mid \mathsf{M}_i \geq 2\}$ and defined as the following set update:

$$\Phi_{\overline{\mathsf{M}}}(X) = \{\phi_W(x) \mid x \in X, W \subseteq [\![n]\!], W \supseteq \{i \in [\![n]\!] \mid i \notin \overline{\mathsf{M}} \vee f_i(x) = 1\}\}.$$

Remark that this definition no longer relies on memory configurations in $\mathbb{N}^n$. Overall, the memory updating mode of BNs can be understood as a particular set of elementary transitions: those where automata not in $\overline{\mathsf{M}}$ or automata that can change from state 0 to 1 are always updated, together with any subset of the others (automata in $\overline{\mathsf{M}}$ that can change from state 1 to 0): automata in $\overline{\mathsf{M}}$ that are decreasing are updated asynchronously, while the others are update in parallel.

Figure 3(left) gives the dynamics generated by the interval updating mode on the BN of Example 2 with $\overline{\mathsf{M}} = \{1\}$.

6 Updating modes going beyond (non-)elementary transitions

BNs are widely used to model dynamics of biological systems, notably implying gene regulation. Gene regulation is a dynamical biological process that involves numerous mechanisms and

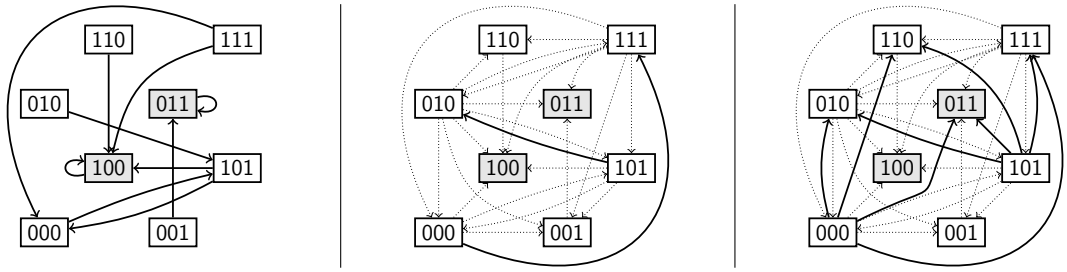

Figure 3 Memory dynamics with $\overline{\mathsf{M}} = \{1\}$ (left), interval dynamics (center), and MP dynamics (right) of the BN of Example 2. In these two latter, the elementary transitions are dotted and loops are omitted.

entities among which some of them, like RNAs and proteins, have specific influences that depend on their concentration. In other terms, the regulation process in its whole admits significant quantitative parts. The question arises then of how faithful are Boolean dynamics with respect to the quantitative dynamics. It has been recently underlined in [24] that the elementary and non-elementary transitions of BNs are not complete enough to capture particular quantitative trajectories. With a fixed logic, and starting from similar configurations, the quantitative system shows that an automaton can eventually get activated, whereas the asynchronous dynamics of the BN shows it is impossible.

In this section, we address the set update reformulation of two recently introduced dynamics of BNs which generate transitions that are neither elementary nor non-elementary: they result in set updates Φ where, for some BNs of dimension n and for some configurations $x \in \mathbb{B}^n$, there is $k \in \mathbb{N}$ such that there exists $y \in \Phi^k(\{x\})$ whereas $x \not\rightarrow_{\mathsf{e}}^* y$.

6.1 Interval updating mode

From the concurrency theory, it is known that the execution of 1-bounded contextual Petri nets corresponding to the asynchronous updating mode (known as *steps* semantics) may miss some transitions that can be triggered when considering certain delay of state change [6, 5]. In [7] is proposed a translation of the *interval* semantics of Petri nets to BNs, showing it can predict transitions that are neither elementary nor non-elementary transitions: configurations that are not reachable with the asynchronous mode from a fixed initial configuration x become reachable with *Interval Boolean networks*.

The main principle of the interval dynamics is to decompose the change of state of an automaton, allowing interleaving the update of other automata. For instance, let us assume that automaton i can change from state 0 to 1. In the interval dynamics, we register that i will eventually change to state 1, and then allow the update of other automata, still considering that i is in state 0. In [7], this is applied to any BN of dimension n by an encoding with the fully-asynchronous dynamics of a BN of dimension $2n$: each automaton is split in a *read* and *write* automaton, where the write automaton register the next state of the original automaton, and the read automaton keeps its current state and will eventually copy the value of the write automaton. The dynamics of the original BN is then obtained by projecting the configurations on the read automata.

We provide here below an equivalent formulation as a composition of set updates. Essentially, whenever an automaton i can change of state, we hold it and compute the possible state changes of the automata different than i. Thus, during this evaluation, we have a growing number of held automata, that we denote by L, waiting for their state to be updated. The set update $\Phi_{\mathrm{Int},L}$ extends a given set of configurations with the possible state change of automata not in L. The function $\Phi_{i/L}(x)$ first computes all possible state changes by iterating $\Phi_{\mathrm{Int},L\cup\{i\}}$ until a fixed point, and then

apply the state change for the automaton i on all the resulting configurations.

▶ **Definition 10.** *The* Interval set update Φ_{Int} *of a BN of dimension n is given by* $\Phi_{\mathrm{Int}} = \Phi_{\mathrm{Int},\emptyset}$ *where*

$$\begin{aligned} \Phi_{\mathrm{Int},L}(X) &= X \cup \{y \in \Phi_{i/L}(x) \mid x \in X, i \in [\![n]\!], i \notin L, f_i(x) \neq x_i\} \\ \Phi_{i/L}(x) &= \{y^{\overline{i}} \mid y \in \Phi^{\omega}_{\mathrm{Int},L\cup\{i\}}(\{x\})\} \end{aligned}$$

The interval updating mode preserves the fixed points of f: for any configuration $x \in \mathbb{B}^n$, $\Phi_{\mathrm{Int}}(\{x\}) = \{x\}$ if and only if $f(x) = x$. Moreover, one can prove that it includes all the elementary transitions: for any configuration $x \in \mathbb{B}^n$, $\Phi_{\mathsf{e}}(\{x\}) \subseteq \Phi_{\mathrm{Int}}(\{x\})$.

▶ **Example 11.** Figure 3 shows the transitions generated by the interval updating mode on the BN of Example 2. Notice that there is a path from 000 to 111, which does not exist in the asynchronous dynamics (Figure 2). Indeed, let us partially compute $\Phi_{\mathrm{Int},\emptyset}(\{000\}) = \{000\} \cup \Phi_{1/\emptyset}(000) \cup \Phi_{3/\emptyset}(000)$.

Let us focus on the interval update of automaton 1 with $\Phi_{1/\emptyset}(000)$, which requires computing all the iterations of $\Phi_{\mathrm{Int},\{1\}}(\{000\})$. The first iteration gives $\Phi_{\mathrm{Int},\{1\}}(\{000\}) = \{000\} \cup \Phi_{3/\{1\}}(000) = \{000, 001\}$, then $\Phi^2_{\mathrm{Int},\{1\}}(\{000\}) = \{000, 001\} \cup \Phi_{2/\{1\}}(001) = \{000, 001, 011\} = \Phi^{\omega}_{\mathrm{Int},\{1\}}(\{000\})$.

Finally, we get $\Phi_{1/\emptyset}(000) = \{100, 101, 111\}$. Thus, $111 \in \Phi_{\mathrm{Int}}(\{000\})$.

6.2 Most Permissive updating mode

Most Permissive Boolean networks (MPBNs) have been designed to capture all automata updates that could occur in any quantitative refinement of the BN. We will come back more formally to this notion later in this section.

The main feature of MPBNs is to abstract all the possible interaction thresholds between automata. Consider the case whenever the state of an automaton i is used to compute the state of two distinct automata j and k, and assume that i is increasing from 0: during its increase, there are times when i may be high enough for trigger a state change of j but not (yet) high enough for k. This can be illustrated on a concrete biological example, the so-called *Incoherent Feed-forward loop of type 3* [20]: a BN f of dimension 3 with

$$f_1(x) = 1 \qquad f_2(x) = x_1 \qquad f_3(x) = \neg x_1 \wedge x_2.$$

Starting from the configuration 000, the asynchronous updating mode predict only the following non-reflexive transitions: $000 \to_{\mathsf{a}} 100 \to_{\mathsf{a}} 110$. Notice that in this case, the interval updating mode results in the same transitions. However, it has been observed experimentally [27] and in quantitative models [19, 26] that depending on reaction kinetics, one can actually activate transiently the automata 3. Essentially, the idea is that during the increase of the state of automaton 1, there is period of time where 1 is high enough so 2 can consider it active (x_1 true) but 3 still considers it inactive (x_1 false). Then, the state of automaton 2 can increase, and so do the state of automaton 3. This activation of 3 cannot be predicted with BN updating modes defined so far, whereas the logic encoded by f is correct.

Without introducing any parameter, MPBNs capture these additional dynamics by accounting for all possible thresholds ordering, for all updates that can happen between a switch of a Boolean state. In some sense, the MP updating mode abstracts both the quantitative domain of automata and the duration of state changes. Their original definition [24] is based on the introduction pseudo *dynamic* states, namely increasing and decreasing. An automaton can change from 0 to increasing whenever it can interpret the state of the other automata so that its local function is satisfied. Once in increasing state, it can change to the state 1 without any condition, or to the decreasing state whenever it can interpret the state of other automata so that is local function is

not satisfied. Whenever an automaton is in a dynamic state, the automata can freely interpret its state as either 0 or 1. Remark that the possible interpretations of the MP configurations always result in a hypercube (a set of automata fixed to a Boolean value, and the others free).

Here, we show that the MP dynamics can be expressed in a more standard way by the means of composition of set updates. A first stage consists in *widening* all the elementary set updates to compute all the possible interpretations of automata changing of state. The widening is defined using the function $\nabla : 2^{\mathbb{B}^n} \to 2^{\mathbb{B}^n}$ which computes the vertices of the smallest hypercube containing the given set of configurations. For instance, $\nabla(\{01,10\}) = \{00,01,10,11\}$. Given a set of automata W, the widening set update $\Phi_{W,\nabla} : 2^{\mathbb{B}^n} \to 2^{\mathbb{B}^n}$ applies this operator on the results of the elementary set update, or equivalently with the fully-asynchronous set update, on the automata of W (Subsection 5.1). This widening is re-iterated until a fixed point. Then, a *narrowing* $\Lambda_W : 2^{\mathbb{B}^n} \to 2^{\mathbb{B}^n}$ filters the computed configurations X to retain only those where the states of automata in W can be computed with f from X.

▶ **Definition 12.** *The* Most Permissive set update Φ_{MP} *of a BN of dimension n is given by*

$$\Phi_{\mathsf{MP}}(X) = \bigcup_{W \subseteq [\![n]\!]} \Lambda_W \circ \Phi^{\omega}_{W,\nabla}(X)$$

where, for any $X \subseteq \mathbb{B}^n$ and any $W \subseteq [\![n]\!]$:

$$\nabla(X) = \{x \in \mathbb{B}^n \mid \forall i \in [\![n]\!], \exists y \in X : x_i = y_i\}, \quad (1)$$

$$\Phi_{W,\nabla}(X) = \nabla(X \cup \{\phi_i(x) \mid x \in X, i \in W\}), \quad (2)$$

$$\Lambda_W(X) = \{x \in X \mid \forall i \in W, \exists y \in X : x_i = f_i(y)\}. \quad (3)$$

▶ **Example 13.** Figure 3 shows the dynamics generated by the MP updating mode on the BN of Example 2. With the BN network f introduced at the beginning of this section, we obtain:

$$\Phi_{\{1,2,3\},\nabla}(\{000\}) = \nabla(\{000,100\}) = \{000,100\}$$

$$\Phi^2_{\{1,2,3\},\nabla}(\{000\}) = \nabla(\{000,100\} \cup \{110\}) = \{000,100,010,110\}$$

$$\Phi^3_{\{1,2,3\},\nabla}(\{000\}) = \nabla(\{000,1000,010,110\} \cup \{011\}) = \mathbb{B}^n$$

$$\Lambda_{\{1,2,3\}}(\mathbb{B}^n) = \{100,101,110,111\}$$

Thus, $111 \in \Phi_{\mathsf{MP}}(\{000\})$, whereas $000 \not\rightarrow_{\mathsf{e}}^* 111$ and $111 \notin \Phi_{\mathrm{Int}}(\{000\})$.

Let us now list some basic properties of the MP updating mode:

1. MP preserves the fixed points of f: for any configuration $x \in \mathbb{B}^n$, $f(x) = x$ if and only if $\Phi_{\mathsf{MP}}(x) = \{x\}$.
2. MP subsumes elementary transitions: $\rightarrow_{\mathsf{e}} \subseteq \delta(\Phi_{\mathsf{MP}})$.
3. MP transition relation is transitive and reflexive: $\Phi_{\mathsf{MP}} = \Phi^2_{\mathsf{MP}}$.
4. (by 2 and 3) MP transition relation subsumes non-elementary transitions: $\rightarrow_{\mathsf{e}}^* \subseteq \delta(\Phi_{\mathsf{MP}})$.
5. (by 4 and the example) there exist BNs f such that the MP transition relation is strictly larger than non-elementary transitions, i.e., there exist $x, y \in \mathbb{B}^n$ such that $y \in \Phi_{\mathsf{MP}}(\{x\})$ but $x \not\rightarrow_{\mathsf{e}}^* y$.

In [24], it has been demonstrated that MP dynamics of a BN f form a correct abstraction of the dynamics of any quantitative model being a *refinement* of f. A quantitative model F can be defined as a function mapping discrete or continuous configurations to the derivative of the state of automata. Then, F is a refinement of f if and only if the derivative of automaton i is strictly positive (resp. negative) in a given quantitative configuration z only if there is a binarization $\tilde{z}$ of z so that $f_i(\tilde{z}) = 1$ (resp. 0). It has also been proven to be minimal for the abstraction of asynchronous discrete models. Moreover, the complexity for deciding the existence of a path between two

configurations as well as deciding whether a configuration belongs to a limit set is respectively in P^{NP} and in $coNP^{coNP}$ in general and in P and in coNP for locally monotonic BNs (each local function is monotonic with respect to a specific component-wise ordering of configurations), in contrast with the other updating modes where these problems are PSPACE-complete.

7 Discussion

By extending to non-deterministic updates modeled as set updates, we can reformulate in a unified manner a range of BN dynamics introduced in the literature with ad-hoc definitions, and for which the usual deterministic updates seem not expressive enough. These reformulations bring a better understanding and comparison of dynamics as more classical BN updating modes. Moreover, they allow envisioning new families of updating modes as variations of the one presented here. For instance, the given MP set update allows to readily define restrictions of it: similarly to the block-sequential updating mode, one could parameterize the MP set update to only consider particular sequences of sets of automata to update. One could also consider different narrowing operators and different manners to compose them with the widening, with the goal of reducing the set of generated transitions.

On the one hand, these set updates foster the definitions of totally new kind of updating modes. On the other hand, they raise the question of a potential upper limit on which transitions could be considered as valid, or at least reasonable.

On *reasonable* set updates

Of course, from a purely theoretical standpoint, any set updates which is mathematically correct is reasonable but, if we consider set updates in a context of modeling, some constraints need to be taken in account. This second standpoint is the one on which is based the following discussion. Indeed, as evoked in the introduction of this paper, BNs are a classical mathematical model in systems biology. They are notably widely used to model genetic regulation networks, in which their use rests for instance on the fact that their limit sets model real observable "structures" such as differentiated cellular types (fixed points), or specific biological paces (limit cycles). In this sense, a basic criterion would be that an updating mode for a BN f is admissible only if the fixed points of f are fixed points of the generated dynamics as well. This criterion would allow capturing the fundamental property of fixed point stability of dynamical system theory. For instance, let us consider the set update $\Phi_\top(X) = \mathbb{B}^n$: clearly, the set of fixed points of the generated dynamical system is always empty, and thus do not include those of f whenever f has at least one fixed point. Therefore, such a set update does not appear satisfying.

Now, let us discuss about set updates which would give sets larger than MP for some singleton configuration set $\{x\}$. First, what about defining a widening operator larger than ∇? For any set of automata W and for any configuration x, remark that $\Phi^{\omega}_{W,\nabla}(\{x\}) = Y$ is the smallest hypercube containing x verifying for each automata $i \in W$ that for any configuration $y \in Y$, if $f_i(y) \neq x_i$, then there exists a configuration $z \in Y$ with $z_i \neq x_i$. Thus, an automaton in W is either fixed to its state in x, or it has been computed with its local function from at least one configuration from a smaller hypercube. Therefore, a widening operator ∇' verifying for some $X \subseteq \mathbb{B}^n$, $\nabla'(X) \supsetneq \nabla(X)$ implies that the state of at least one configuration is not computed using f on X. Now, what about a less stringent narrowing operator. Let us consider a configuration $y \in \Phi_{W,\nabla}(\{x\}) = Y$ for some set of automata W, but $y \notin \Lambda_W(Y)$. This implies that there exists an automaton $i \in W$ such that $\forall z \in Y$, $y_i \neq f_i(z)$, i.e., y_i cannot be computed by f_i from X. Overall, a set update giving configuration sets strictly larger than the MP update implies that for some configurations, the state of at least one automaton is not computed using its local function.

Simulations by deterministic updates

A perspective of the work presented in this paper focuses on simulations of BNs evolving with non-deterministic updates by BNs evolving with deterministic updates. A first natural way is by following a classical determinization of the dynamics. Indeed, one can encode any set of configurations in $\mathbb{B}^n$ as one configuration in $\mathbb{B}^{2^n}$. Let us consider such an encoding $c : 2^{\mathbb{B}^n} \to \mathbb{B}^{2^n}$ where, for all $x \in \mathbb{B}^n$, $c(X)_x = 1$ if $x \in X$, otherwise $c(X)_x = 0$ (we slightly abuse notations here, by specifying a vector index by its binary representation). Now, it is clear that for any set update $\Phi : 2^{\mathbb{B}^n} \to 2^{\mathbb{B}^n}$ of a BN f of dimension n, one can define a BN g such that for all sets of configurations $X \subseteq \mathbb{B}^n$, $g(c(X)) = c(\Phi(X))$. This encoding is complete in the sense that any transition generated by Φ is simulated in (g, p). But these simulations are nothing else but a brute-force encoding in which we get rid of the transition relation by increasing exponentially the state space. Moreover, with this deterministic encoding, the structure of the transition relation of $(f, \mu = \Phi)$ is lost, which make much more difficult characterizing dynamical features of (f, μ) such as its limit sets for instance.

Actually, a fundamental matter here lies in the concept of simulation at stake here: we are interested in intrinsic simulations which go far beyond the classical concepts of encoding or simulation. Indeed, intrinsic simulations aim at conserving dynamical structures in addition to operated computations. So, one of the first question to answer would consist in defining formally different kinds of intrinsic simulations. Nevertheless, firstly, consider the following intrinsic simulation: a dynamical system (f, μ) simulates another (g, μ') if $\mathcal{D}_{(g,\mu')}$ is a subgraph of $\mathcal{D}_{(f,\mu)}$. With this rather simple definition, it is direct to state that, with a and $\overline{\mathsf{M}}$ the asynchronous and memory updating modes respectively, for any BN f, (f, a) simulates $(f, \overline{\mathsf{M}})$. Some natural questions related to BNs updated with memory are the following:

- Are there BNs whose dynamics obtained according to $\overline{\mathsf{M}}$ remains deterministic, whatever $\overline{\mathsf{M}}$?
- If so, what are their properties and what are the equivalent deterministic updating modes?

To go further, consider the MP updating mode. It is direct that (f, μ) does not simulate (f, MP), except for very particular f. Let us now consider a more general intrinsic simulation: a dynamical system (f, μ) simulates another (g, μ') if $\mathcal{D}_{(g,\mu')}$ is a graph obtained from $\mathcal{D}_{(f,\mu)}$ thanks to edge deletions, and vertex shortcuts. A lot of promising questions arise from this, in particular related to $\overline{\mathsf{M}}$ and MP updating modes, among which for instance:

- Let per be a deterministic periodic updating mode. How can $(f, \overline{\mathsf{M}})$ be simulated by (g, per)? The answer is known for $\mathsf{per} = \mathsf{p}$ [14], but it seems pertinent to find a generalization to deterministic periodic updating modes, and even more general deterministic updating modes.
- Intuitively, any (f, MP) might be simulated by (g, a), where f and g are BNs and the dimension of g is greater than that of f. But how many automata need to be added to g depending on the dimension of f?

All answers, even partial or negative, will bring a better understanding of updating modes and BNs, which would lead to pertinent further development in both BN theory and their application in systems biology.

References

1 Julio Aracena. Maximum number of fixed points in regulatory Boolean networks. *Bulletin of Mathematical Biology*, 70:1398–1409, 2008.

2 Julio Aracena, Eric Fanchon, Marco Montalva, and Mathilde Noual. Combinatorics on update digraphs in Boolean networks. *Discrete Applied Mathematics*, 159:401–409, 2011.

3 Julio Aracena, Eric Goles, Andres Moreira, and Lilian Salinas. On the robustness of update schedules in Boolean networks. *Biosystems*, 97:1–8, 2009.

4 Florian Bridoux, Caroline Gaze-Maillot, Kévin Perrot, and Sylvain Sené. Complexity of limit-cycle problems in Boolean networks. In *Proceedings of the International Conference on Current Trends in*

Theory and Practice of Informatics (SOFSEM'21), volume 12607 of *Lecture Notes in Computer Science*, pages 135–146. Springer, 2021.

5 Thomas Chatain, Stefan Haar, Juraj Kolčák, Loïc Paulevé, and Aalok Thakkar. Concurrency in Boolean networks. *Natural Computing*, 19(1):91–109, 2020. doi:10.1007/s11047-019-09748-4.

6 Thomas Chatain, Stefan Haar, Maciej Koutny, and Stefan Schwoon. Non-atomic transition firing in contextual nets. In *Applications and Theory of Petri Nets*, volume 9115 of *Lecture Notes in Computer Science*, pages 117–136. Springer, 2015. doi:10.1007/978-3-319-19488-2_6.

7 Thomas Chatain, Stefan Haar, and Loïc Paulevé. Boolean Networks: Beyond Generalized Asynchronicity. In *Cellular Automata and Discrete Complex Systems (AUTOMATA 2018)*, volume 10875 of *Lecture Notes in Computer Science*, pages 29–42, Ghent, Belgium, 2018. Springer. doi:10.1007/978-3-319-92675-9_3.

8 Michel Cosnard and Jacques Demongeot. On the definitions of attractors. In *Proceedings of the International Symposium on Iteration Theory and its Functional Equations*, volume 1163 of *Lecture Notes in Mathematics*, pages 23–31. Springer, 1985.

9 Jacques Demongeot, Eric Goles, Michel Morvan, Mathilde Noual, and Sylvain Sené. Attraction basins as gauges of the robustness against boundary conditions in biological complex systems. *PLoS One*, 5:e11793, 2010.

10 Jacques Demongeot, Mathilde Noual, and Sylvain Sené. Combinatorics of Boolean automata circuits dynamics. *Discrete Applied Mathematics*, 160:398–415, 2012.

11 Jacques Demongeot and Sylvain Sené. About block-parallel Boolean networks: a position paper. *Natural Computing*, 19:5–13, 2020.

12 A. Elena. *Robustesse des réseaux d'automates booléens à seuil aux modes d'itération. Application à la modélisation des réseaux de régulation génétique*. PhD thesis, Université Grenoble 1 – Joseph Fourier, 2009.

13 Françoise Fogelman, Eric Goles, and Gérard Weisbuch. Transient length in sequential iteration of threshold functions. *Discrete Applied Mathematics*, 6:95–98, 1983.

14 Eric Goles, Fabiola Lobos, Gonzalo A. Ruz, and Sylvain Sené. Attractor landscapes in Boolean networks with firing memory: a theoretical study applied to genetic networks. *Natural Computing*, 19:295–319, 2020.

15 Eric Goles, Pedro Montealegre, and Martín Ríos-Wilson. On the effects of firing memory in the dynamics of conjunctive networks. In *Proceedings of the International Workshop on Cellular Automata and Discrete Complex Systems (AUTOMATA'19)*, volume 11525 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2019.

16 Eric Goles and Mathilde Noual. Block-sequential update schedules and Boolean automata circuits. In *Proceedings of the International Workshop on Cellular Automata and Discrete Complex Systems (AUTOMATA'10)*, pages 41–50. Discrete Mathematics and Theoretical Computer Science, 2010.

17 Eric Goles and Lilian Salinas. Comparison between parallel and serial dynamics of Boolean networks. *Theoretical Computer Science*, 396:247–253, 2008.

18 Alex Graudenzi and Roberto Serra. A new model of genetic networks: the gene protein Boolean network. In *Proceedings of the Workshop italiano su vita artificiale e calcolo evolutivo (WIVACE'09)*, pages 283–291. World Scientific, 2009.

19 Shuji Ishihara, Koichi Fujimoto, and Tatsuo Shibata. Cross talking of network motifs in gene regulation that generates temporal pulses and spatial stripes. *Genes to Cells*, 10(11):1025–1038, 2005. doi:10.1111/j.1365-2443.2005.00897.x.

20 S. Mangan and U. Alon. Structure and function of the feed-forward loop network motif. *Proceedings of the National Academy of Sciences*, 100(21):11980–11985, 2003. doi:10.1073/pnas.2133841100.

21 John Milnor. On the concept of attractor. *Communications in Mathematical Physics*, 99:177–185, 1985.

22 Mathilde Noual. *Updating automata networks*. PhD thesis, École normale supérieure de Lyon, 2012.

23 Mathilde Noual and Sylvain Sené. Synchronism vs asynchronism in monotonic Boolean automata networks. *Natural Computing*, 17:393–402, 2018.

557 **24** Loïc Paulevé, Juraj Kolčák, Thomas Chatain, and Stefan Haar. Reconciling qualitative, abstract, and
558 scalable modeling of biological networks. *Nature Communications*, 11(1), 2020. doi:10.1038/
559 s41467-020-18112-5.
560 **25** François Robert. *Discrete iterations: a metric study*, volume 6 of *Springer Series in Computational*
561 *Mathematics*. Springer, 1986.
562 **26** Guillermo Rodrigo and Santiago F. Elena. Structural discrimination of robustness in transcriptional
563 feedforward loops for pattern formation. *PLOS ONE*, 6(2):e16904, 2011. doi:10.1371/journal.pone.
564 0016904.
565 **27** Yolanda Schaerli, Andreea Munteanu, Magüi Gili, James Cotterell, James Sharpe, and Mark Isalan.
566 A unified design space of synthetic stripe-forming networks. *Nature Communications*, 5(1), 2014.
567 doi:10.1038/ncomms5905.