Synthesis and Simulation of Ensembles of Boolean Networks for Cell Fate Decision

Stéphanie Chevalier*[1], Vincent Noël*[2], Laurence Calzone[2], Andrei Zinovyev[2], and Loïc Paulevé[3]

[1] LRI, CNRS, UMR8623, Univ. Paris-Saclay, France
[2] Institut Curie, INSERM, U. PSL, Mines ParisTech, France
[3] Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, F-33400 Talence, France

Abstract. The construction of models of biological networks from prior knowledge and experimental data often leads a multitude of candidate models. Devising a single model from them can require arbitrary choices, which may lead to strong biases in subsequent predictions.
We introduce here a methodology for a) synthesizing Boolean model ensembles satisfying a set of biologically relevant constraints and b) reasoning on the dynamics of the ensembles of models. The synthesis is performed using Answer-Set Programming, based on the approach introduced by us earlier. Here we extend the synthesis to account for solution diversity and for universal constraints on reachable fixed points, enabling accurate specification of desired dynamics.
The sampled models are then simulated and the simulation results are aggregated through averaging or can be analyzed as a multi-dimensional distribution. The obtained results allow an interpretation at the level of cell population and take into account its heterogeneity.
We illustrate our approach on a previously published Boolean model of a molecular network regulating the cell fate decisions in cancer progression. It appears that the ensemble-based approach to Boolean modelling brings new insights on the variability of the synergistically interacting mutation effect with respect to the propensity of a cancer cell to metastasize.

1 Introduction

The ability to derive one single model from observations of a biological system usually faces arbitrary choices, sometimes referred to as *art*.

Computational models of molecular interaction networks are usually built from data related to the structure of the biological system, such as known interactions; and data related to its dynamics, such as measurements of gene expressions or proteins activity at different time and/or conditions. However, despite huge advances in experimental technologies, observations of the biological processes stay very scarce, either in terms of temporal resolution, number of observed entities, synchronisation between measure points, or variety of experimental conditions. Combined with complex structures for molecular interactions, the model

* co-first authors

engineering problem in this case appears to be largely under-specified, leading to (too) many potential candidate models.

Boolean Networks (BNs), and logical models in general, are widely adopted for the modelling of signalling pathways and gene and transcription factors networks [5,27,3], as they are not demanding for the exact knowledge of the quantitative parameters of the selected molecular interactions. With BNs, the activity of components is caricatured to "off" and "on", and their evolution is computed according to logical rules (e.g., gene 1 can be active only whenever its activators 2 and 3 are active). However, in practice, biological data still let open a multitude of candidate BNs. Thus, arbitrary modelling choices have to be made, e.g., by prioritizing certain logics between regulators or by preferring smallest/largest models, which may introduce biases in subsequent model predictions.

In this paper, we present an approach aiming at reducing modelling biases by constructing and reasoning on dynamics of *ensembles* of BNs. The idea of ensemble modelling has recently gained momentum with machine learning, in particular with the random forest approach. By analogy, we constitute ensembles of models representative of the whole multitude of Boolean models compatible with network architecture and dynamical properties. They are then simulated asynchronously and the simulations are aggregated through averaging. The obtained results allow an interpretation at the level of cell population and take into account its potential heterogeneity.

In the literature, ensembles of *random* BNs have been employed to show emerging properties of families of BNs sharing properties related to their architecture or logic rules [13,15]. In [20], ensembles of BNs sharing a network architecture are used to assess dynamical properties of qualitative differential equations. In these works, ensembles are not built from dynamical properties such as reachability or attractor, contrary to our approach which is focused on ensembles of models satisfying a set of constraints coming either from the biological knowledge or experimental data.

Our ensembles are built using the formal synthesis of BNs with the logic framework of Answer-Set Programming [1]. We extend our prior work on BNs synthesis from reachability and attractor properties with Most Permissive semantics [2] by enabling universal properties on (reachable) fixed points and by considering different network perturbation settings. Then, we use heuristics to drive the ASP solver in different regions of the solution space in order to sample ensembles of BNs representating the diversity of the comprehensive model set.

Dynamics of resulting ensembles are then explored by means of stochastic simulations in order to quantify the propensities of reachable attractors, possibly subject to network perturbations. To that aim, we extended the simulator MaBoSS [21] to support ensembles of BNs as input.

We illustrate our approach on a model of molecular pathways regulating tumour invasion and migration [4]. We sampled ensembles of BNs sharing the same network architecture as the original model and constrained by the dynamical properties related to attractor reachability. Then, as in the original study, we evaluated the shift of reachable phenotypes caused by an epistatic interaction

$$f_1(x) := \neg x_2$$
$$f_2(x) := \neg x_1$$
$$f_3(x) := \neg x_1 \wedge x_2$$

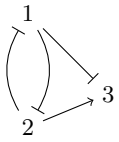

Fig. 1. Example of Boolean network f and its influence graph $G(f)$ where positive edges are with normal tip and negative edges are with bar tip.

between mutations in model genes (gain of Notch function and loss of p53 function). It appears that, contrary to the initial single model analysis, the ensemble approach reveals a potential variability in the effectiveness of the double mutant to enhance the metastasis potential.

2 Background

2.1 Boolean Networks

A *Boolean network* (BN) of dimension n is a function

$$f : \mathbb{B}^n \to \mathbb{B}^n \tag{1}$$

where $\mathbb{B} := \{0, 1\}$. For all $i \in [n]$, $f_i : \mathbb{B}^n \to \mathbb{B}$ denotes the *local function* of the i-th component. A vector $x \in \mathbb{B}^n$ is called a *configuration* of the BN f. The set of components which differ between two configurations $x, y \in \mathbb{B}^n$ is denoted by $\Delta(x, y) := \{i \in [n] \mid x_i \neq y_i\}$.

A BN f is said *locally monotonic* whenever each of its local functions is monotonic (this does not imply f monotonicity). Intuitively, when expressing the local functions using propositional logic, local monotonicity imposes that a variable always appears with the same sign in a minimal normal form.

Fig. 1 is an example of locally-monotonic BN with $n = 3$.

Mutations In the following, we will consider the analysis of a BN f subject to some *permanent* perturbations of its components, that we refer to as mutations, being either a *gain of function* (GoF; locked to 1) and *loss of function* (LoF; locked to 0). A mutation is specified by a couple (i, v), where $i \in [n]$ is a component and $v \in \mathbb{B}$ is its forced value. Given a BN f and a set of mutations $M \subseteq [n] \times \mathbb{B}$, we denote by f/M the mutated BN, where, for each $i \in [n]$, $(f/M)_i(x) := v$ if $(i, v) \in M$, and $(f/M)_i(x) := f_i(x)$ otherwise.

Influence graph For each $i \in [n]$, f_i typically depends on a small subset of components of the BN. The *influence graph* summarizes these dependencies with a positive (resp. negative) edge from node j to i if there are configurations in which the sole increase of j would strictly increase (resp. decrease) the value of f_i. A node can have both positive and negative influences on i, indicating that f_i is non-monotonic. Remark that different BNs can have the same influence graph. Fig. 1 (right) shows the influence graph of the BN example.

Definition 1. *Given a BN f of dimension n, its* influence graph *$G(f)$ is a directed graph $([n], E_+, E_-)$ with* positive *and* negative *edges such that $(j,i) \in E_+$ (resp. $(j,i) \in E_-$) iff $\exists x, y \in \mathbb{B}^n$ s.t. $\Delta(x,y) = \{j\}$, $x_j < y_j$, and $f_i(x) < f_i(y)$ (resp. $f_i(x) > f_i(y)$). The influence graph $\mathcal{G} = ([n], E_+, E_-)$ is a* subgraph *of $\mathcal{G}' = ([n], E'_+, E'_-)$, denoted by $\mathcal{G} \subseteq \mathcal{G}'$, iff $E_+ \subseteq E'_+$ and $E_- \subseteq E'_-$.*

2.2 BN Semantics

From a configuration $x \in \mathbb{B}^n$, semantics of BNs specify how to compute the next possible configurations. One of the most classical semantics is the fully-asynchronous (often simply called asynchronous), where only one component i is updated at a time (to the value $f_i(x)$). It can be defined as a binary relation $\xrightarrow[\text{a1}]{f}$ between configurations:

Definition 2 (Fully-asynchronous semantics).

$$\forall x, y \in \mathbb{B}^n, \quad x \xrightarrow[\text{a1}]{f} y \text{ iff } \exists i \in [n] : \Delta(x,y) = \{i\} \wedge y_i = f_i(x) \ .$$

We write $\rho^f_{\text{a}}(x) := \{y \in \mathbb{B}^n \mid x \xrightarrow[\sigma]{f}{}^* y\}$ the set of configurations in transitive relation with x, with $\xrightarrow[\text{a}]{f}{}^*$ the reflexive and transitive closure of $\xrightarrow[\text{a1}]{f}$.

However, as demonstrated in [18], the fully-asynchronous semantics of BNs, as the synchronous and (general) asynchronous, are not faithful abstractions of quantitative systems: they can both introduce spurious behaviours (as expected with qualitative models) and miss others.

The *Most Permissive* (MP) semantics of BNs [18] offers the guarantees to not preclude any behaviour realisable in any quantitative refinement of the model, thus providing a formal over-approximation of dynamics. Moreover, the abstraction is minimal: any behaviour it predicts is realisable by a quantitative refinement of the BN using the asynchronous semantics. Importantly, the complexity for deciding main dynamical properties is considerably lower than with (a)synchronous semantics, as we will mention in the next subsection.

MPBNs can be defined by the means of hypercubes (partially) closed by f.

Definition 3 (Hypercube). *An* hypercube *h of dimension n is a vector in $(\mathbb{B} \cup \{*\})^n$. The set of its associated configurations is denoted by $c(h) := \{x \in \mathbb{B}^n \mid \forall i \in [n], h_i \neq * \Rightarrow x_i = h_i\}$.*

Given two hypercubes $h, h' \in (\mathbb{B} \cup \{*\})^n$, h is smaller *than h' if and only if $\forall i \in [n], h'_i \neq * \Rightarrow h_i = h'_i$.*

Definition 4 (K-closed hypercube). *Given a subset of components $K \subseteq [n]$, an hypercube $h \in (\mathbb{B} \cup \{*\})^n$ is* K-closed *by f whenever for each configuration $x \in c(h)$, for each component $i \in K$, $h_i \in \{*, f_i(x)\}$.*
It is minimal *whenever no different K-closed hypercube is smaller than it.*

An hypercube $[n]$-closed by f is also known as a *trap space* [14].

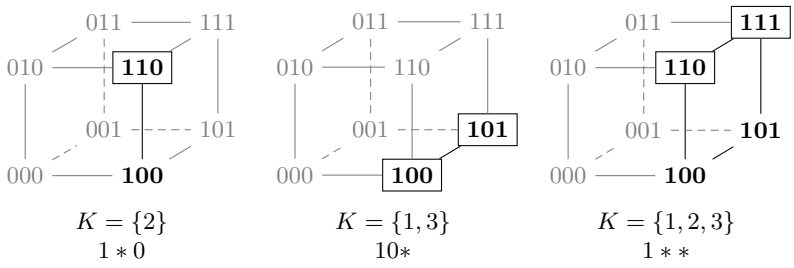

Fig. 2. Examples of smallest K-closed hypercubes containing the configuration 100 for the BN f of dimension 3 defined by $f_1(x) := 1$, $f_2(x) := x_1$, $f_3(x) := x_1 \wedge \neg x_3$. Configurations belonging to the hypercube are in bold; these verifying the MP reachability property are boxed. The hypercube $11*$ is only one which is closed by f and minimal.

Example 1. Let us consider the BN $f : \mathbb{B}^3 \to \mathbb{B}^3$ with $f_1(x) := \neg x_2$, $f_2(x) := \neg x_1$, et $f_3(x) := \neg x_1 \wedge x_2$. The hypercube $01*$ is closed by f, with $c(01*) = \{010, 011\}$. The hypercube $1*0$ is the smallest hypercube $\{2,3\}$-closed by f containing 110; it is not closed by f, nor the smallest hypercube $\{2,3\}$-closed by f containing 100.

Starting from a configuration $x \in \mathbb{B}^n$, the MP semantics allows transitions towards any configuration y which is present in at least one smallest K-closed hypercube h containing x, for some $K \subseteq n$, and so that the state of each component $i \in K$ of y can be computed by f_i from a configuration of h.

Definition 5 (Most-Permissive semantics). *Given a BN f of dimension n and two configurations $x, y \in \mathbb{B}^n$, $y \in \rho^f_{\mathrm{mp}}(x)$ if and only if there exists $K \subseteq [n]$ such that the smallest K-closed hypercube h containing x verifies (1) $y \in c(h)$, and (2) $\forall i \in K$, there exists a configuration $z \in c(h)$ such that $f_i(z) = y_i$.*

Fig. 2 gives examples of computations of ρ^f_{mp}.

A way to interpret the MP semantics is to see the components free in an hypercube as being in the course of changing of state; and other components can independently consider them either as 0 or 1. This aims at abstracting the missing information at Boolean level on the thresholds of activation or inhibition between components: while the quantitative value of component u progressively increases, at a given time it can be high enough to activate a component (i.e., 1) but not yet high enough to activate another one (i.e., 0). These *dynamic* states are overlooked by asynchronous semantics, making it an incorrect over-approximation of quantitative systems, contrary to the MP semantics [18].

2.3 Dynamical properties

In the following, we will focus on two main dynamical properties of BNs: *reachability* which relates to the existence of trajectories between two configurations, and *attractors* which relates to long-run behaviours by identifying the smallest sets of configurations closed by reachability.

Definition 6 (Reachability). *Given two configurations $x, y \in \mathbb{B}^n$ of a BN f with semantics σ, y is* reachable *from x whenever $y \in \rho^f_\sigma(x)$.*

Definition 7 (Attractor). *A non-empty set of configurations $A \subseteq \mathbb{B}^n$ is an* attractor *of the BN f with semantics σ whenever $\forall x \in A, \rho^f_\sigma(x) = A$.*
When $A = \{x\}$ for some $x \in \mathbb{B}^n$, we say that x is a fixed point*.*

With MP semantics, attractors match with minimal trap spaces. With fully-asynchronous semantics, deciding if $y \in \rho^f_{\text{a1}}(x)$ or if x belongs to an attractor are both PSPACE-complete problems. With MP semantics, deciding if $y \in \rho^f_{\text{mp}}(x)$ is PTIME if f is locally-monotonic and P^{NP} otherwise; deciding if x belongs to an attractor is coNP if f is locally-monotonic and $\text{coNP}^{\text{coNP}}$ otherwise. Deciding if there exists a fixed point is NP-complete with both semantics [18].

Notice the following relations between MP and fully-asynchronous semantics:

- $x \in \mathbb{B}^n$ is a fixed point with MP semantics if and only if it is a fixed point with full-asynchronous semantics (iff $f(x) = x$);
- $y \in \mathbb{B}^n$ is reachable from $x \in \mathbb{B}^n$ with the fully-asynchronous semantics *only if* it is reachable with MP semantics ($\rho^f_{\text{a1}}(x) \subseteq \rho^f_{\text{mp}}(x)$);
- the number of attractors with MP semantics is less than or equal to the number of attractors with fully-asynchronous semantics.

2.4 Answer-Set Programming

Answer Set Programming (ASP; [1,9]) is a declarative approach to solving combinatorial satisfaction problems. It is close to SAT (propositional satisfiability) [16] and known to be efficient for enumerating solutions of NP problems comprising up to tens of millions of variables, while providing a convenient language for specifying the problem. We give a very brief overview of ASP syntax and semantics that we use in the next sections; see [9] for more details.

An ASP program is a Logic Program (LP) being a set of logical rules with first order logic predicates of the form:

```
1 a_0 ← a_1 , ..., a_n , not a_{n+1} , ..., not a_{n+k} .
```

where a_i are (variable-free) atoms, i.e., elements of the Herbrand base, which is built from all the possible predicates of the LP. The Herbrand base is built by instantiating the LP predicates with the LP terms (constants or elements of the Herbrand universe).

Essentially, such a logical rule states that when all $a_1, \ldots, a_n$ are true and none of $a_{n+1}, \ldots, a_{n+k}$ can be proven to be true, then a_0 has to be true as well. Whenever a_0 is $\bot$ (false), the rule, also called integrity constraint, becomes:

```
2 ←a_1 , ..., a_n , not a_{n+1} , ..., not a_{n+k} .
```

Such a rule is satisfied only if the right hand side of the rule is false (at least one of $a_1, \ldots, a_n$ is false or at least one of $a_{n+1}, \ldots, a_{n+k}$ is true). On the other hand, $a_0 \leftarrow \top$ (a_0 is always true) is abbreviated as a_0. A solution (answer set) is a *stable* Herbrand model, that is, a minimal set of true atoms where all the logical rules are satisfied. For instance, consider the following program:

```
3 a.
4 b ← a.
5 d ← a, c.
```

It has for unique solution {`a, b`}: indeed, whereas $\{a, b, d\}$ does not contradict the rules, `d` is not a fact and cannot be derived from a rule; so it is not stable.

ASP allows using variables (starting with an upper-case) instead of terms/predicates: these *template* declarations will be expanded prior to the solving. We also use the notation `a(X): b(X)` which is satisfied when for each `b(X)` true, `a(X)` is true. If any term follows such a condition, it is separated with `;`.

ASP can express *disjunctive* logic programs [17], by the means of disjunctions in the rule head (";"-separated atoms):

```
6 a; b ← body.
```

Such a disjunctive rule implies that solutions are subset minimal: an answer set is solution only if none of its subsets is itself a solution [8]. For instance, let's consider the disjunction:

```
7 a; b; c.
```

The interpretation $I = \{a, b\}$ is a model but not minimal: both interpretations $\{a\}$ and $\{b\}$ are smaller than I and satisfy the rule. Hence I is not a solution. As showed in [7], the complexity of problems addressed with ASP can be extended thanks to disjunctive rules up to 2QBF, i.e. a two quantification levels Boolean formula ($\forall x \exists y.\phi$ or $\exists y \forall x.\phi$ where ϕ is a quantifier-free propositional formula). Indeed, 2QBF can be reduced to the problem of verifying the existence of an answer set of a disjunctive ASP program.

3 BN Synthesis from Architecture and Dynamical Properties

We formulate the problem of BN synthesis as a Boolean satisfiability problem encoded in Answer-Set Programming. With this approach, we leverage a priory knowledge and experimental data as constraints on the graph topology and the dynamical properties of the models (under the MP semantics). Our method and its implementation were introduced in [2], with several constraint types such as existence of a fixed point or irreversibility of a bifurcation. In biological applications, these constraints match well the observed properties of cell populations evolving towards mutually exclusive phenotypes.

This approach has been extended in the present work to universal properties (also called global properties). Such a property not only ensures that a described behaviour is in the system dynamics, but it also ensures it's the only possible behaviour. To illustrate, let's compare the meaning of existential and universal properties given a list of experimentally observed cell fates. An existential constraint guarantees that at least one attractor of the model dynamics match with each cell fate. A universal constraint ensures that every model attractor matches with at least one of the cell fate.

A universal property involves by nature universal quantifiers. ASP can address formulas implying one level of universal quantifier (i.e., of the form $\exists x \forall y : P(x, y)$) thanks to a technique presented in [7]. To explore a set of values and check the respect of a property for each, it ingeniously uses a disjunctive rule and a saturation on the same term. A disjunctive rule actually implies the subset minimality semantics. This minimality ensures an answer set is solution only if none of its subsets is itself a solution [8]. Hence, saturate the answer set with the predicates subjected to the disjunction cleverly exploits this minimality: the solver is forced to explore all subsets of these predicates to check that none of them are smaller solutions.

3.1 Universal constraints on fixed points

We exploit this *saturation technique* for ensuring universal constraints on the fixed points or on fixed points reachable from a given configuration. We describe here the ASP rules for the universal fixed point constraint, which ensures that all the fixed points of the BN are compatible with a given set of markers (observations). To that purpose, we let the solver deduce a configuration z by the disjunctive rule:

```
8 cfg(z,N,-1) ; cfg(z,N,1) ← node(N).
```

The predicate template `cfg(X,N,V)` assigns the value `V` to the literal `N` in the configuration `X`. Through the above rule, a set of node values is thus constituted to define a configuration z, with the predicate `cfg(z,N,_)` submitted to the subset minimality semantics. To respect the desired property, each configuration z is either not a fixed point ($f(z) \neq z$) or has the same component states than the ones expressed in a dedicated predicate. A configuration is not a fixed point whenever at least one of its component can change of state:

```
9 mcfg(z,N,V) ← cfg(z,N,V).
10 valid ← cfg(z,N,V) ; eval(z,N,-V).
```

`mcfg(X,N,V)` predicate template leads to the evaluation of the configuration `X` given the Boolean rules of the network [2]. The reachable values are then stored in the predicate `eval(X,N,V)`.

In the other case, if a configuration is a fixed point, it has to have same component states than those specifed by an observation `X` marked by the predicate `is_universal_fp(X)`.

```
11 valid ← cfg(z,N,V):obs(X,N,V); is_universal_fp(X).
```

Observe in l.10 and l.11 that each time an assignment is in agreement with the desired property, a predicate `valid` is deduced. `valid` is used to saturate the answer set on the predicates submitted to the disjunctive rules, predicates which represent the node values in the configuration z:

```
12 cfg(z,N,-V) ← cfg(z,N,V), valid.
```

Thus, when `valid` is deduced, the answer set contains all possible component values for z. According to the subset minimality semantics, the solver is then forced to ensure that there is no sub-answer set. And the only way to find such a smaller answer-set is to find a z from which `valid` cannot be deduced, i.e., which is a counter-example to the universal property: in that case, l.13 eliminates the answer set:

```
13 ← not valid.
```

A variant of this constraint enables to limit the research of fixed points to those reachable from a configuration of interest. For this purpose, the fact `is_universal_fp(X,S)` describes that from `S`, only fixed points with the characteristics of `X` are reachable. Such predicates are combinable to define a set of fates reachable from `S`. The encoding of this variant contains a third way to deduce the predicate `valid`: the non-reachability of the configuration z from `S`.

Our implementation also offers to consider mutations, which can be combined with reachability and with universal constraints on reachable fixed points to leverage observations about cell fates in different mutation conditions.

3.2 Synthesis problem

Synthesis requires (i) an influence graph to delimit the interactions that can be included in the models and (ii) the dynamical properties of the behaviours that have to be reproduced. For modelling the tumor invasion and migration as in [4], the dynamical properties are cell fate observations given conditions. These fates are described by sets of markers (i.e. a set of values for some nodes of the network) which constitutes partial observations of genes activity. In the Boolean network dynamics, these observations are related to reachable attractors.

A (partial) observation o of a configuration of dimension n is specified by a set of couples associating a component to a Boolean value: $o \subseteq [1]n \times \mathbb{B}$, assuming there is no $i \in [1]n$ such that $\{(i,0),(i,1)\} \subseteq o$.

Formally, the synthesis problem we tackle is the following.
Given

- an influence graph $\mathcal{G} = \{[n], E_+, E_-)$
- p partial observations $o^1, \ldots, o^p$
- sets FP, UFP and UA of indices of observations
- sets PR, URFP and URA of couples of indices of observations: $\mathsf{URFP} \subseteq [p]^2$

find a locally-monotonic BN f of dimension n such that

- $G(f) \subseteq \mathcal{G}$,
- there exist p configurations $x^1, \ldots, x^p$ such that:
  - (observations) $\forall m \in [p], \forall (i,v) \in o^m, x^m_i = v$,
  - (positive reachability) $\forall (m,m') \in \mathsf{PR}, x^{m'} \in \rho^f_{\text{mp}}(x^m)$,
  - (fixpoints) $\forall m \in \mathsf{FP}, f(x^m) = x^m$,
  - (universal fixed point) $\forall z \in \mathbb{B}^n, f(z) = z \Rightarrow \exists m \in \mathsf{UFP} : \forall (i,v) \in o^m, z_i = v$;

- (universal reachable fixed point) $\forall z \in \mathbb{B}^n, f(z) = z \Rightarrow \exists (x, s) \in \mathsf{URFP} : z \notin \rho^f_{\text{imp}}(s) \vee \forall (i, v) \in x, z_i = v$;

Remark that such a problem can be non-satisfiable.

Our encoding also offers constraints related to the absence of paths between configurations (*negative reachability*) and to *trap space* where a set of components have a fixed state matching with a given observation [2]. Moreover, one can optionally impose that the influence graph of f is equal to the input $\mathcal{G}$.

Our implementation avoids redundancy in the models by enumerating only among non-equivalent BNs (i.e., their values differ for at least one configuration). This is achieved by using a canonical representation of Boolean functions in disjunctive normal form with a total ordering between clauses. In total, our encoding generates $O(ndk^2)$ atoms and $O(nd^2k^2)$ rules, where d is the in-degree of nodes in the influence graph, and k is a fixed bound on the number of clauses of Boolean functions. Whenever k is set to $\binom{d}{\lfloor d/2 \rfloor}$, the complete set of solutions can be enumerated.

3.3 Sampling the diversity of all solutions

The whole set of constraints, comprising the domain of admissible BNs and the dynamical properties they should satisfy, is represented by a single logic program expressed in ASP, such that each solution corresponds to a distinct BN.

Whereas the enumeration of ASP solutions is known to be efficient, typical solvers will enumerate solutions by slightly varying parts of a firstly identified one. Thus, a partial enumeration will very likely give a set of solutions which are all look alike, e.g., with only on the Boolean function of only one component.

Inspired by [19], we tweak heuristics of the solver `clingo`[10] to stir it towards distant solutions: at each solution, we randomly select a subset of variables assignments and ask the solver to avoid them in the next iterations. At the cost of enumeration speed, this allows sampling ensembles of *diverse* BNs.

4 Stochastic Simulations of Ensembles of BNs

4.1 Continuous-time Boolean modelling

We first recall the continuous-time Markov chain interpretation of BNs introduced in [22].

Considering a single BN f of dimension of n, we represent the state evolution by a Markov process $s : t \to s(t)$ defined on $t \in I \subset \mathbb{R}$ applied on the network state space, with I the simulation interval. This process is defined by:

1. Its initial condition:
$$P[s(0) = x], \quad \forall x \in \mathbb{B}^n$$
2. Its conditional probabilities (of a single condition):
$$P[s(t) = y | s(t') = x], \quad \forall x, y \in \mathbb{B}^n, \forall t', t \in I, t' < t$$

In continuous time, these conditional probabilities are defined as transition rates[24]: $\rho(x \to y)(t) \in [0, \infty]$. Because we want a generalization of the fully-asynchronous Boolean dynamics, transition rates $\rho(x \to y)$ are non-zero only if $x \xrightarrow[\text{a1}]{f} y$, i.e., a single component $i \in [n]$ is changing of value. In that case, each local function $f_i(x)$ is replaced by two functions $R_i^{up/down}(x) \in [0, \infty]$.

The transition rates are defined as follows: if i is the node that differs from x and y, then

$$\rho(x \to y) = R_i^{up}(x), \quad if \quad x_i = 0$$

$$\rho(x \to y) = R_i^{down}(x), \quad if \quad x_i = 1$$

where R_i^{up} corresponds to the activation rate of node i, and R_i^{down} corresponds to the inactivation rate of node i. Therefore, the continuous Markov process is completely defined by all these $R^{up/down}$ and an initial condition.

To explore the probability space of this Markov process, we use Gillespie algorithm[11]. This algorithm produces a set of realizations or stochastic trajectories of the Markov process. The set of stochastic trajectories represents the given Markov process in the sense that these trajectories can be used to compute probabilities. A finite set of these trajectories is produced, then, from this finite set, probabilities are estimated.

To relate continuous time probabilities to real processes, an observable time window δt is defined. A discrete time $\tau \in \mathbb{N}$ stochastic process $s(\tau)$ can be extracted from the continuous time Markov process:

$$P[s(\tau) = x] \equiv \frac{1}{\delta t} \int_{\tau \delta t}^{(\tau+1)\delta t} P[s(t) = x] dt$$

For each trajectory j, we compute the time for which the system is in state x in the window $[\tau\delta t, (\tau + 1)\delta t]$, and divide it by δt. We obtain an estimate of $P[s(\tau) = x]$ for trajectory j, i.e. $\hat{P}_j[s(\tau) = x]$. Then to compute the estimate of a set of trajectories, we compute the average over j of all $\hat{P}_j[s(\tau) = x]$.

This simulation algorithm is implemented in the MaBoSS C++ simulation software [21][4].

4.2 Lifting to ensembles of BNs

To simulate the ensemble of BNs, we first choose a total number of stochastic trajectories M. We generate M/k stochastic trajectories for each model, and compute the average $\hat{P}_k[s(\tau) = x]$ for all models k. We then compute the average over k of all $\hat{P}_k[s(\tau) = x]$, to obtain the $P[s(\tau) = x]$ for the ensemble of boolean networks. We also keep the option to export the individual probability distributions $\hat{P}_k[s(\tau) = x]$ to allow us analyzing the composition of the ensemble. The approach results in time-series of the probability for each observed state. The case study hereafter focuses on steady state analysis. This imposes to simulate

[4] `https://maboss.curie.fr`, `https://github.com/colomoto/pyMaBoSS`

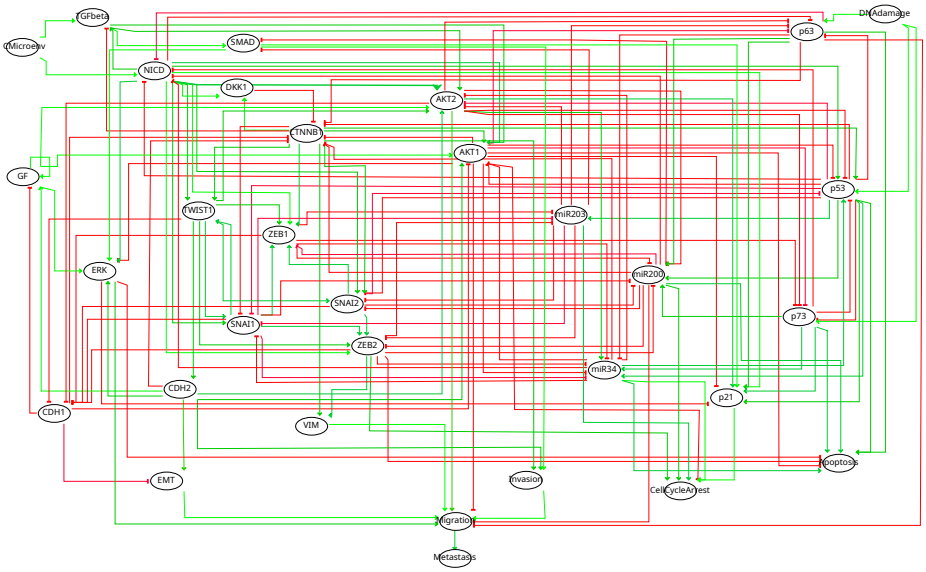

Fig. 3. Influence graph of Cohen's model relating 32 nodes with 159 edges, where positive edges are in green and negative in red.

the ensemble long enough to reach stationarity, requiring a preliminary analysis. We can then study the proportion of each attractor for our ensemble.

This ensemble simulation algorithm has been implemented in C++ and is now included in MaBoSS.

5 Case Study on Cell Fate Decision Modelling

5.1 Background Model

We illustrate our ensemble modelling approach on a published model of cell fate decision leading to the early events of the metastasis or to cell death through apoptosis [4]. Initial triggers, such as DNA damage or micro-environmental cues, and the activity of some genes or proteins participating in the process affect the final decision. The signalling pathway involve TGFbeta, WNT, beta-catenin, p53 and its homologs, selected miRNA, and transcription factors of the epithelial to mesenchymal (EMT) transitions. Fig. 3 shows the influence graph of the BN.

The functions of the BN, we refer to as "Cohen's model", have been designed manually so the simulations fit with experimental data related to stable phenotypes under different single mutations. Then, the initial publication explored the synergy between mutants that led to metastatic phenotypes.

5.2 Single Model Analysis

We first reproduced part of the analysis of [4] on the original Cohen's model by computing the propensities of attractors reachable from 4 possible initial conditions, where all nodes are inactive, except miRNAs that are active, and the 2 nodes modelling DNA damage and micro-environment cues that are free. We considered the wild-type condition (Fig. 4(a)) with no mutations, and the double-mutant of p53 LoF and Notch GoF (Fig. 4(b)).

The wild-type model has 9 fixed points that each correspond to one of the 4 identified physiological phenotypes: Apoptosis, EMT, Metastasis (or equivalent to Migration) and Homeostatic State (HS). The double-mutant then shows exclusively the Metastasis phenotype.

5.3 Ensemble Analysis

Synthesis In order to show the impact of alternative Boolean functions, we synthesised ensembles of BNs that share the exact same influence graph as Cohen's model and reproduce the desired dynamics. We synthesized two ensembles of 1,000 diverse BNs each, where we disallowed having cyclic attractors. The first ensemble ensures only the wild-type (WT) behaviour, meaning that all the fixed points match with one of the 4 physiological phenotypes, and each physiological phenotype is reachable from at least one of the initial condition. The second ensemble adds further constraints related to the single mutations of p53 LoF, which should show the same behaviour as WT, and Notch GoF, where only 2 of the WT phenotypes and a third different one should be observed[5].

Ensemble Simulations With the same settings as with Cohen's model, we performed stochastic simulations of the two synthesized ensembles, with uniform activation and de-activation rates. Whereas the WT behaviours look similar, although with some differences in propensities of phenotypes (Fig. 4(c,e)), the double-mutant shows a much less contrasted picture that with the single model analysis: While Migration becomes the most likely outcome, several other phenotypes are observed, suggesting a potential variability of the effect of the double-mutation. Interestingly, even the single mutant constraints of the second ensemble are not sufficiently restrictive to guarantee the behaviour observed in Cohen's model.

Variability of propensities of phenotypes In order to study ensemble composition, we want to analyze the steady-states probabilities for each model. Depending on the results we might have a lot of visited states, which bring a dimensionality issue. We choose to represent these results using Principal Component

[5] Code, data, and notebooks at `https://gist.github.com/pauleve/13cf6e7dc79a90e663451f87d1860a6e`; Synthesis has been performed on 36-cores CPUs @ 2.6Ghz with 192Go of RAM; first ensemble was generated at a rate of 5s/model/CPU; second ensemble was generated at a rate of 3min/model/CPU

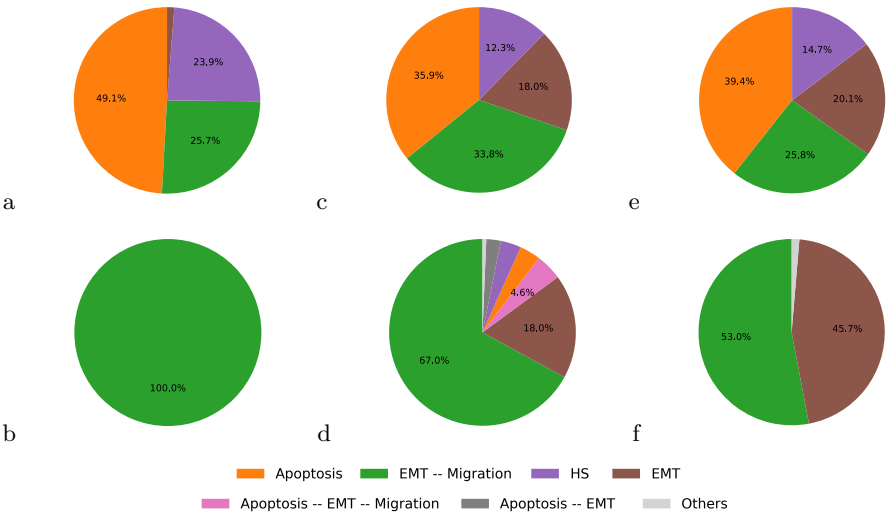

Fig. 4. Simulations results for phenotypes propensities in Cohen's model (a,b), ensemble from WT constraints (c,t), and ensemble from WT and single mutants constraints (e,f), in wild-type condition (a,c,e) and double-mutant p53 LoF/Notch GoF (b,d,f).

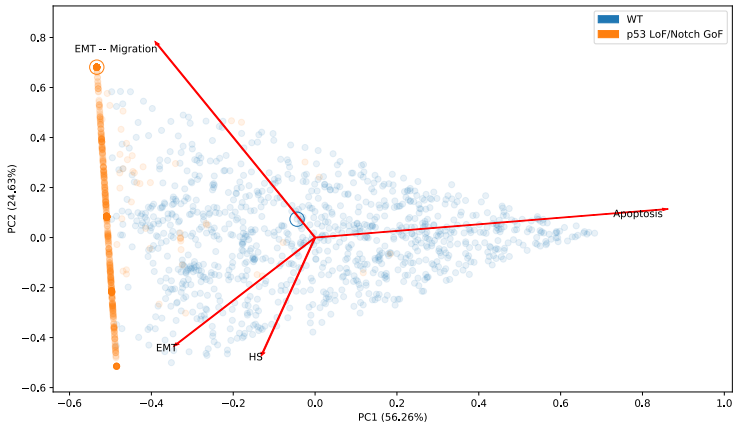

Fig. 5. PCA representation of steady state distribution of each individual model of the ensemble from WT and single mutants constraints. Each point represents the result of one model simulation (blue one are from WT simulations, orange one from p53 LoF/Notch GoF). Large blue and orange circles highlight the position of the original single Cohen's model simulation. The triangular pattern of the distribution comes from the fact that the phenotype probabilities are located in the n-dimensional simplex.

Analysis (PCA)[25], which allows us to visualize the distribution of attractor's proportions in a reduced number of dimensions.

We apply PCA to the probability distribution of each individual model within the ensemble from WT and single mutants constraints, allowing us to represent their respective probability distributions (Fig. 5). The first component, representing 56% of the observed variance, shows a negative correlation between apoptotic and EMT phenotypes. The second component, representing 24% of observed variance, shows a negative correlation between EMT without Migration and EMT with Migration. The distribution of the ensemble's probability distributions is diverse, illustrating the performance of the all possible model diversity sampling. The p53 LoF/Notch GoF double mutant shows a shift towards EMT and/or Migration phenotypes, away from Apoptotic phenotypes. The alignment of models on the top-left corresponds to models which don't show any apoptotic phenotypes ($\sim$ 96% of the models).

6 Conclusion

The synthesis of BNs from network architecture and dynamical constraints [23,6,26,12] can lead to a multitude of admissible solutions.

In this work, we employed Answer-Set Programming to sample ensembles of diverse BNs, all possessing the same topology and satisfying the same set of dynamical constraints. We significantly extend the previously described methodology with the new type of biologically relevant universal constraints.

Our synthesis framework enables specifying existence and absence of reachability properties between (partial) configurations of the BN, existence of fixed points and cyclic attractors matching with observations, and universal properties on the fixed points and reachable fixed points; all these properties can be parametrized by mutation settings.

The dynamics of ensembles is explored by stochastic simulations using the new Ensemble MaBoSS simulator, which is introduced here for the first time. The ensemble-based simulations are used for computing and comparing propensities of reachable attractors under different mutations or their combinations. The result of an ensemble-based simulation represents a multidimensional distribution of the vectors of attractor probabilities, which can be aggregated by computing its mean point. Moreover, the multi-variate variance of the distribution can be explored, e.g. by applying the standard machine learning methods such as Principal Component Analysis, which can lead to the insights about the diversity of possible modelling scenarios compatible with available biological knowledge and the experimental data.

As we illustrate on a biological case study, ensemble modelling has the potential of improving the robustness of predictions by accounting for potential model variability and uncertainty.

References

1. Baral, C.: Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press (2003)
2. Chevalier, S., Froidevaux, C., Paulevé, L., Zinovyev, A.: Synthesis of Boolean Networks from Biological Dynamical Constraints using Answer-Set Programming. In: 31st International Conference on Tools with Artificial Intelligence. ICTAI, IEEE, Portland, Oregon, United States (2019)
3. Clarke, M.A., Fisher, J.: Executable cancer models: successes and challenges. Nature Reviews Cancer (apr 2020). https://doi.org/10.1038/s41568-020-0258-x
4. Cohen, D.P.A., Martignetti, L., Robine, S., Barillot, E., Zinovyev, A., Calzone, L.: Mathematical modelling of molecular pathways enabling tumour cell invasion and migration. PLoS Comput Biol **11**(11), e1004571 (2015). https://doi.org/10.1371/journal.pcbi.1004571
5. Collombet, S., van Oevelen, C., Sardina Ortega, J.L., Abou-Jaoudé, W., Di Stefano, B., Thomas-Chollier, M., Graf, T., Thieffry, D.: Logical modeling of lymphoid and myeloid cell specification and transdifferentiation. Proceedings of the National Academy of Sciences **114**(23), 5792–5799 (2017). https://doi.org/10.1073/pnas.1610622114
6. Dorier, J., Crespo, I., Niknejad, A., Liechti, R., Ebeling, M., Xenarios, I.: Boolean regulatory network reconstruction using literature based knowledge with a genetic algorithm optimization method. BMC Bioinformatics **17**(1), 410 (2016). https://doi.org/10.1186/s12859-016-1287-z
7. Eiter, T., Gottlob, G.: On the computational cost of disjunctive logic programming: Propositional case. Annals of Mathematics and Artificial Intelligence **15**(3), 289–323 (Sep 1995). https://doi.org/10.1007/BF01536399
8. Eiter, T., Ianni, G., Krennwallner, T.: Answer Set Programming: A Primer, pp. 40–110. Springer Berlin Heidelberg, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03754-2_2
9. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Answer Set Solving in Practice. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan and Claypool Publishers (2012)
10. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Clingo = ASP + control: Preliminary report. CoRR **abs/1405.3694** (2014)
11. Gillespie, D.T.: A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. Journal of computational physics **22**(4), 403–434 (1976)
12. Goldfeder, J., Kugler, H.: BRE:IN - a backend for reasoning about interaction networks with temporal logic. In: Computational Methods in Systems Biology, pp. 289–295. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-31304-3_15
13. Kauffman, S.: A proposal for using the ensemble approach to understand genetic regulatory networks. Journal of Theoretical Biology **230**(4), 581–590 (oct 2004). https://doi.org/10.1016/j.jtbi.2003.12.017
14. Klarner, H., Bockmayr, A., Siebert, H.: Computing maximal and minimal trap spaces of boolean networks. Natural Computing **14**(4), 535–544 (2015). https://doi.org/10.1007/s11047-015-9520-7
15. Krawitz, P., Shmulevich, I.: Basin entropy in boolean network ensembles. Physical Review Letters **98**(15) (apr 2007). https://doi.org/10.1103/physrevlett.98.158701

16. Lin, F., Zhao, Y.: ASSAT: Computing answer sets of a logic program by SAT solvers. Artificial Intelligence **157**(1), 115–137 (2004)
17. Lobo, J., Minker, J., Rajasekar, A.: Foundations of disjunctive logic programming. MIT press (1992)
18. Paulevé, L., Kolčák, J., Chatain, T., Haar, S.: Reconciling qualitative, abstract, and scalable modeling of biological networks. bioRxiv (2020). https://doi.org/10.1101/2020.03.22.998377
19. Razzaq, M., Kaminski, R., Romero, J., Schaub, T., Bourdon, J., Guziolowski, C.: Computing diverse boolean networks from phosphoproteomic time series data. In: Computational Methods in Systems Biology, pp. 59–74. Springer International Publishing (2018). https://doi.org/10.1007/978-3-319-99429-1_4
20. Schwieger, R., Siebert, H.: Graph representations of monotonic boolean model pools. In: Computational Methods in Systems Biology, pp. 233–248. Springer International Publishing (2017). https://doi.org/10.1007/978-3-319-67471-1_14
21. Stoll, G., Caron, B., Viara, E., Dugourd, A., Zinovyev, A., Naldi, A., Kroemer, G., Barillot, E., Calzone, L.: MaBoSS 2.0: an environment for stochastic boolean modeling. Bioinformatics **33**(14), 2226–2228 (2017). https://doi.org/10.1093/bioinformatics/btx123
22. Stoll, G., Viara, E., Barillot, E., Calzone, L.: Continuous time boolean modeling for biological signaling: application of gillespie algorithm. BMC systems biology **6**(1), 116 (2012)
23. Terfve, C., Cokelaer, T., Henriques, D., MacNamara, A., Goncalves, E., Morris, M.K., Iersel, M.v., Lauffenburger, D.A., Saez-Rodriguez, J.: CellNOptR: a flexible toolkit to train protein signaling networks to data using multiple logic formalisms. BMC Systems Biology **6**(1), 133 (2012). https://doi.org/10.1186/1752-0509-6-133
24. Van Kampen, N.G.: Stochastic processes in physics and chemistry, vol. 1. Elsevier (1992)
25. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemometrics and intelligent laboratory systems **2**(1-3), 37–52 (1987)
26. Yordanov, B., Dunn, S.J., Kugler, H., Smith, A., Martello, G., Emmott, S.: A method to identify and analyze biological programs through automated reasoning. Systems Biology and Applications **2** (2016)
27. Zañudo, J.G., Steinway, S.N., Albert, R.: Discrete dynamic network modeling of oncogenic signaling: Mechanistic insights for personalized treatment of cancer. Current Opinion in Systems Biology **9**, 1–10 (2018). https://doi.org/10.1016/j.coisb.2018.02.002