Relationship between the Reprogramming Determinants of Boolean Networks and their Interaction Graph

Hugues Mandon, Stefan Haar, Loïc Paulevé

Relationship between the Reprogramming Determinants of Boolean Networks and their Interaction Graph

Hugues Mandon[1], Stefan Haar[2], Loïc Paulevé[1]

[1] LRI UMR 8623, Univ. Paris-Sud – CNRS, Université Paris-Saclay, France
[2] LSV, ENS Cachan, INRIA, CNRS, Université Paris-Saclay, France

Abstract. In this paper, we address the formal characterization of targets triggering cellular trans-differentiation in the scope of Boolean networks with asynchronous dynamics. Given two fixed points of a Boolean network, we are interested in all the combinations of mutations which allow to switch from one fixed point to the other, either possibly, or inevitably. In the case of existential reachability, we prove that the set of nodes to (permanently) flip are only and necessarily in certain connected components of the interaction graph. In the case of inevitable reachability, we provide an algorithm to identify a subset of possible solutions.

1 Introduction

In the field of regenerative medicine, an emerging way to treat patients is to reprogram cells, leading, for instance, to tissue or neuron regeneration. Such a challenge has become realistic after first experiments have shown that some of the cell fate decisions can be reversed [15]. Whereas the cells go through several multipotent states before reaching a differentiated state, the differentiation process can be inversed, producing induced pluripotent stem cells (iPSCs) from an already differentiated cell. By using a distinct differentiation path, this allows to "transform" the type of a cell. Alternatively, it is also possible to directly perform a trans-differentiation without necessarily going (back) through a multipotent state [9,7].

In the aforementioned work, the de- and trans-differentiation has been achieved by targeting specific genes, that we refer to as *Reprogramming Determinants* (RDs), through the mediation of their transcription factors [15,6].

The computational prediction of RDs requires to assess multiple features of the cell dynamics and the reprogramming strategy, such as the impact of the kind of perturbations (persistent versus temporary) and of their order; the nature of targeted cell type (differentiated/pluripotent), and the desired inevitability of their reachability (fidelity); the nature and duration of the triggered cascade of regulations (efficiency); and finally, the RD robustness with respect to initial state heterogeneity among cell population, and with respect to uncertainties in the computational model.

So far, no general framework allows to efficiently encompass those features to systematically predict best combinations of RDs in distinct cellular reprogramming events.

In this paper, we address the identification of RDs from *Boolean Networks* (BNs) which model the dynamics of gene regulation and signalling networks. The state of the components (or nodes) of the networks are represented by Boolean variables, and the state changes are specified by Boolean functions which associate the next state of nodes, given the (binary) state of their regulators [16,2]. BNs are well suited for an automatic reasoning on large biological networks where the available knowledge is mostly about activation and inhibition relations[1]. Such activation/inhibition relations between components form a signed directed graph, that we refer to as the *Interaction Graph*.

In this work, we make the assumption that the differentiated cellular states correspond to the *attractors* of the dynamics of the computational model, i.e., the long-run behaviours. In the scope of BNs, those attractors can be of two kinds: either a single state (referred to as a fixed point), or a terminal cyclic behaviour.

The relationship between the IG of BNs and the number of their attractor has been extensively studied [2,13,14]. However, little work exists on the characterization of the perturbations which trigger a change of attractor. Currently, most of RDs prediction are performed using statistical analysis on expression data in order to rank candidate transcription factors [3,12,10]. Whereas based on network models, those approaches do not allow to derive a complete set of solution for the reprogramming problem. In [6], the authors developed a heuristic to derive candidate RDs from a pure topological analysis of the interaction graph: the RDs are selected only in positive cycles that have different values in the started and target fixed points. However, there is no guarantee that the derived RDs can actually lead to a change of attractor in the asynchronous dynamics of the Boolean networks, and neither that the target fixed point is the only one reachable. Finally, [8] gives a formal characterization of RDs subject to temporal mutations which trigger a change of attractor in the synchronous semantics of conjunctive Boolean networks.

Contribution This work relies on model checking and reachability analysis, that have been proved useful and successful in previous studies[1,11].

Given a BN, all of whose attractors are fixed points, given an initial fixed point and a target fixed point, we provide a characterization of the candidate RDs (set of nodes) with respect to the interaction graph and for two settings of cellular reprogramming:

- with a permanent perturbation of RDs, the target fixed point becomes reachable in the asynchronous dynamics of the BN;
- with a permanent perturbation of RDs, the target fixed point is the sole reachable attractor in the asynchronous dynamics of the BN.

For the first case, we prove that all the RDs are distributed among particular strongly connected components of the interaction graph, and we give algorithms

to determine them in both settings. In the second case, we prove that only some of them are distributed among strongly connected components of the interaction graph. We provide an algorithm to identify possible combination of permanent perturbations leading to inevitable reachability of the target fixed point. Whereas the algorithm may miss some solutions, all returned solutions are correct.

Outline Section 2 gives the definitions and basic properties of BNs and of their asynchronous dynamics. The formalization of the BN reprogramming problem with permanent perturbations of nodes is established in Sect. 3. Section 4 states the main results on the characterization of RDs with respect to the interaction graph of BNs. An algorithm to enumerate all RDs by exploiting this characterization is given in Sect. 5. Finally, 6 discusses the results and sketches future work.

Notations

Given a finite set I, 2^I is the power set of I, $|I|$ the cardinality. Given a positive integer n, $[n] = \{1, \ldots, n\}$.

Given a Boolean state $x \in \{0,1\}^n$ and set of indexes $I \subset [n]$, $\bar{x}^I$ is the state where $\bar{x_i}^I = x_i$ if $i \notin I$ and $\bar{x_i}^I = 1 - x_i$ if $i \in I$. Similarly, given $x, y \in \{0,1\}^n$, $x_{[x_I = y_I]}$ denotes the state where for all $i \in I$, $(x_{[x_I=y_I]})_i = y_i$ and for all $i \notin I$, $(x_{[x_I=y_I]})_i = x_i$

2 Background

In this section, we give the formal definition of Boolean networks, their interaction graph and transition graph in the asynchronous semantics. Finally, we recall the main link between their attractors and the positive cycles in their interaction graph.

2.1 Definitions

Boolean Network (BN): A BN is a finite set of Boolean variables, each of them having a Boolean function. This function is a logical Boolean function depending from the network's variables and determining the next state of the variable.

Definition 1 (Boolean Network (BN)). *A Boolean Network is a function f such that:*

$$\begin{aligned} f : \quad & \{0,1\}^n \to \{0,1\}^n \\ & x = (x_1, ..., x_n) \mapsto f(x) = (f_1(x), ..., f_n(x)) \end{aligned}$$

Example 1. An example of BN of dimension 3 ($n = 3$) is

$$f_1(x) = x_3 \vee (\neg x_1 \wedge x_2)$$
$$f_2(x) = \neg x_1 \vee x_2$$
$$f_3(x) = x_3 \vee (x_1 \wedge \neg x_2)$$

Interaction Graph: To determine the RDs, we rely on a simplification of the interactions between the genes, and of the concentrations. A gene will either be active or inhibited. Gene interactions are simplified likewise, a gene either activates or inhibits another gene, and we ignore time scales. With this in mind, an *interaction graph* (Def.2) can be build: genes are the vertices, and the interactions are the oriented arcs, labelled either + or −, if it is an activation or an inhibition.

Definition 2 (Interaction Graph). *An interaction graph is noted as* $G = (V, E)$*, with* V *being the vertex set, and* E *being the directed, signed edge set,* $E \subset (V \times V \times \{-, +\})$

A cycle between a set of nodes $C \subseteq V$ is said positive (resp. negative) if and only if there is an even (odd) number of negative edges between those nodes.

An interaction graph can also be defined as an abstraction of a Boolean network: the functions are not given and not always known, but if a vertex u is used in the function f_v, there is an edge from u to v, negative if $f_v(x)$ contains $\neg x_u$ and positive if it contains x_u.

Definition 3 (Interaction Graph of a Boolean network ($G(f)$)). *An interaction graph can be obtained from the Boolean network* f*: the vertex set is* $[n]$*, and for all* $u, v \in [n]$ *there is a positive (resp. negative) arc from* u *to* v *if* $f_{vu}(x)$ *is positive (resp. negative) for at least one* $x \in \{0, 1\}^n$ *(For every* u*,* $v \in \{1, ..., n\}$*, the function* f_{vu} *is the discrete derivative of* f_v *considering* u*, defined on* $\{0, 1\}^n$ *by :* $f_{vu}(x) := f_v(x_1, .., x_{u-1}, 1, x_{u+1}, .., x_n) - f_v(x_1, .., x_{u-1}, 0, x_{u+1}, .., x_n)$*).*

Given an interaction graph $G = (V, E)$, and one of its vertex $u \in V$, P_u denotes the set of ancestors of u, i.e., the vertices v for which there exists a path in E from v to u. Similarly, p_u is the set of the parents of u, i.e., $v \in p_u \Rightarrow (v, u, s) \in E$. Furthermore, $G[P_u]$ is the induced subgraph of G with P_u as vertex set.

Fig. 1 gives an example of an interaction graph, which is also equal to $G(f)$, where f is the Boolean network of Ex.1.

Transition Graph: We model the dynamics of a Boolean network f by *transitions* between its states $x \in \{0, 1\}^n$. In the scope of this paper, we consider the *asynchronous semantics* of Boolean networks: a transition updates the value of only one vertex $u \in [n]$. From a single $x \in \{0, 1\}^n$, one has different transitions for each vertex u such that $f_u(x) \neq x_u$. This leads to the definition of the *transition graph* (Def. 4) where vertices are all the possible states $\{0, 1\}^n$, and edges correspond to asynchronous transitions.

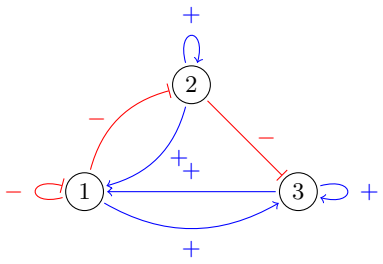

Fig. 1. Interaction graph of Ex.1 A "normal" blue arrow means an activation, and a "flattened" red arrow means an inhibition.

Definition 4 (Transition graph). *The transition graph is the graph having* $\{0,1\}^n$ *as vertex set and the edges set* $\{x \to \bar{x}^{\{u\}} \mid x \in \{0,1\}^n, u \in [n], x_u \neq (f(x))_u\}$. *An existing path from* x *to* y *is noted* $x \to^* y$.

Fig.2 gives the transition graph of the asynchronous dynamics of Boolean network of Ex.1.

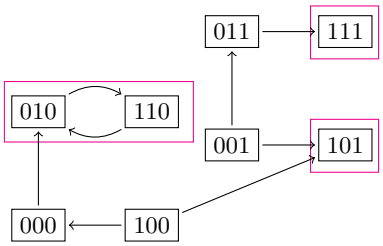

Fig. 2. Transition graph of the Boolean network defined in Ex.1. We use shorter notations, 010 meaning that the node 1 has 0 as value, the node 2 has 1 as value, and the node 3 has 0 as value. The attractors are boxed in magenta.

Attractors, Fixed point : BN's *Attractors* are the terminal strongly connected components of the transition graph, and can be seen as the long-term dynamics of the system. Note that an attractor is always a set of states, but it can contain either multiple distinct nodes, that is the system oscillate between multiple states (*cyclic attractor*) or a unique point, i.e the system stays in the same state (*fixed point*).

Definition 5 (Attractor).

$$S \subseteq \{0,1\}^n \text{ is an attractor} \Leftrightarrow S \neq \emptyset \quad (1)$$
$$\text{and } \forall x \in S,\ \forall y \in \{0,1\}^n \setminus S,\ x \not\to y \quad (2)$$
$$\text{and } \forall x \in S,\ S \setminus x \text{ does not verify (2)} \quad (3)$$

If $|S| = 1$ then S is a fixed point. Otherwise S is a cyclic attractor.

Given a BN f, $\mathrm{FP}(f) \subseteq \{0,1\}^n$ denotes the set of its fixed points ($\forall x \in \mathrm{FP}(f), f(x) = x$).

Example 2. The BN of Ex.1 has 3 attractors that correspond to the 3 terminal strongly connected components of Fig.2: $\{010, 110\}$ (cyclic attractor), $\{101\}$ and $\{111\}$ (fixed points).

2.2 On the link between attractors and the interaction graph

Theorem 1 is a conjecture by René Thomas [16] that has been since demonstrated for Boolean and discrete networks [2,17]: if a Boolean network has multiple attractors then its interaction graph necessarily contains a positive cycle. In the case of multiple fixed points, any pair of fixed point differ at least on a set of nodes forming a positive cycle.

Theorem 1 (Thomas' first rule). *If $G = (V, E)$ has no positive cycles, then f has at most one attractor. Moreover, if f has two distinct fixed points x and y, then G has a positive cycle between vertices $C \subseteq V$ such that $x_v \neq y_v$ for every vertex v in C.*

We can also remark that for a vertex to stay at a value y_v where y is a fixed point, it only needs its ancestors to have the same values as in y.

Remark 1. $\forall y \in \mathrm{FP}(f), \forall u \in [n], \forall z \in \{0,1\}^n$, z verifying $\forall j \in P_u,\ z_j = y_j,$ we have $f_u(y) = y_u = f_u(z)$.

Proof. Let u be a vertex in $[n]$. $f(u)$ only depends of the incoming arcs in u, so it only depends of p_u, which in turn depends on its parents. By induction, $f_u(y)$ only depends of P_u, and so, if $f_u(y) = y_u$ in G, then $f_u(y) = y_u$ in $G[P_u]$. □

3 Formalisation of the BN Reprogramming with Permanent Perturbations

Given two fixed points x and y of Boolean network f, we want to identify sets of nodes, referred to as Reprogramming Determinants (RDs), that when changed in x enable to switch to y. As our theorems rely on the differences between the fixed points, we chose to focus on fixed points solely. Further work will extend, if possible, these theorems and algorithms to all kind of attractors. In the scope of this paper, by "change" we mean permanently set the vertex to a new fixed

value. If we "change" u to 1 (resp. 0), then $f_u(x) = 1$ (resp. 0) for all x. When switching to y (by changing I) is possible, we have two cases : it either means that y is reachable from $x_{[x_I=y_I]}$ (existential reachability, Def. 6), or that y is the only reachable fixed point from $x_{[x_I=y_I]}$ (inevitable reachability, Def. 7). These are two different approaches that we will both consider. To remove the temporal aspect, we make all the changes at the same time (hence $x_{[x_I=y_I]}$, otherwise an order should be visible), and only watch if y is reachable. This also means that there is no indication of how long it takes for y to be reached.

Definition 6 (Existential Reachability). *With the boolean network F, a function ER_F can be defined as $ER_F : 2^{2^{[n]}}$, with $ER_F(x, y) \mapsto v$ where v is the set of all minimal vertex sets I such that $x_{[x_I=y_I]} \to^* y$.*

Definition 7 (Inevitable Reachability). *Similarly, a function $IR_F : 2^{2^{[n]}}$ can be defined as $IR_F(x, y) \mapsto w$ where w is the set of all minimal vertices sets I such as $\forall z \in \{0, 1\}^n$, $x_{[x_I=y_I]} \to^* z \Rightarrow z \to^* y$.*

These two functions will give different results, and have different meanings, as shown in the examble below.

Example 3. Let us consider the BN f of Fig.3 and its transition graph reproduced in Fig.4. f has 4 fixed points: $0000, 0001, 1100$ and 1101. Let $x = 0000$ and $y = 1100$. Fixing the node $\{1\}$ to 1 in x makes y reachable : 1100 (=y) is reachable from $x_{[x_1=1]} = 1000$ with the Boolean network f' defined by $f'_1(x) = 1$ and $f'_2 = f_2$, $f'_3 = f_3$, $f'_4 = f_4$. The transition graph of f', considering the first node being active, corresponds to the left part of the transition graph in Fig.4. One can then remark that y is not the only fixed point reachable: from 1000, 1101 is also reachable. If we also fix the node $\{4\}$ to 0, y is the only reachable fixed point from $x_{[x_1=1,x_4=0]}$ in the Boolean network f'' such that $f''_1(x) = 1$, $f''_2 = f_2$, $f''_3 = f_3$, and $f''_4(x) = 0$.

Therefore, with the previous definitions, $\{1\} \in ER_F(0000, 1100)$ but $\{1\} \notin IR_F(0000, 1100)$; and $\{1, 4\} \in IR_F(0000, 1100)$ but $\{1, 4\} \notin ER_F(0000, 1100)$. Moreover, we also have $\{1, 2\}$ and $\{1, 3\} \in IR_F(0000, 1100)$.

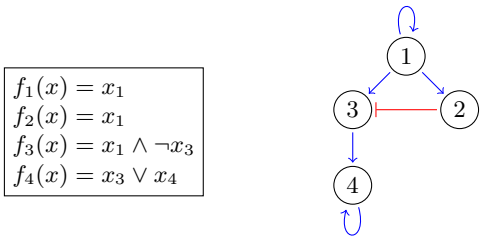

Fig. 3. A BN of dimension 4

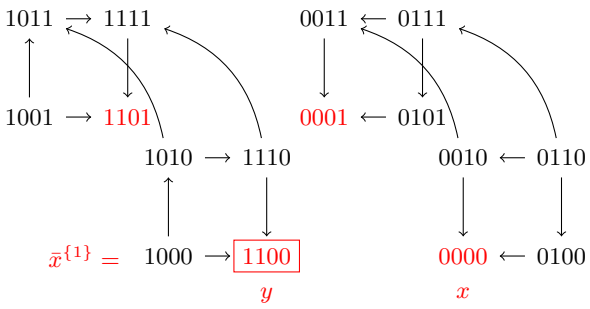

Fig. 4. Transition graph of the BN in Fig.3

4 Reprogramming Determinants and the SCCs of the Interaction Graph

In this section, we show the link between the RDs and the Strongly Connected Components (SCCs) of the interaction graph of the Boolean network f. Our results make the assumption that all the attractors of f are fixed points (no cyclic attractors).

4.1 SCC Ordering

To switch from x to y, we want to change the value of each vertex u that has different values for x and y ($x_u \neq y_u$) and to prevent each vertex v that verifies $x_v = y_v$ from changing value. We know that changing the value of a vertex can have an impact on other vertices, but we also know that it will only impact its descendants.

So, if a vertex has a different value in x and y but none of its ancestors do, then it is necessary to change this vertex. So, to know which vertices need to be changed first, the best way is to order them, with a topological order for example. Of course, if there are loops, an order is impossible to determine, we have to reduce all SCCs to single "super-vertices" to achieve it. In the remaining of this paper, we will consider SCCs which contain at least one positive cycle, because they are known to change between fixed points (Theor.1), we call $\mathcal{O}$ the SCC set that contains all such SCCs. Reducing the graph to its SCCs makes possible to rank them from 1 to k with any topological order, noted $\prec$: for all $i, j \in [k], j > i \Rightarrow \mathcal{O}_j \not\prec \mathcal{O}_i$.

Let C_0 be the set $\{\mathcal{O}_i \in \mathcal{O} \mid \nexists \mathcal{O}_j, \mathcal{O}_j \prec \mathcal{O}_i\}$, and recursively define slices $C_K = \{\mathcal{O}_i \in (\mathcal{O} \setminus \bigcup_{l\in\{1,..,K-1\}} C_l) \mid \nexists \mathcal{O}_j, \mathcal{O}_j \prec \mathcal{O}_i\}$. Given the definition of the slices, for all topological orders, the slice set will be the same. The slices are numbered from 1 to c.

From this order, we know which SCCs need to be impacted, still, SCCs ranked lower in the hierarchy need not be impacted by the change in their ancestors

(see ex.5) The relation $\prec$ only gives an order to make the changes, from which one can determine if further changes are needed.

Example 4. Showing that only using the topological order is not sufficient.

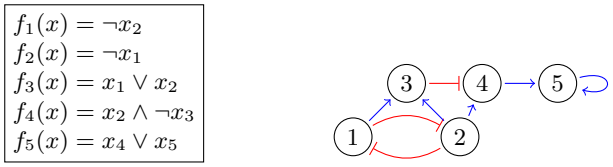

Fig. 5. BN preventing changes in the lower SCC

Any algorithm that only used the topological order without computing the reachable fixed points would not suffice, as the example from Fig.5 shows : the switch from the fixed point 01100 to 10101 would be computed by just modifying $\{1\}$, but in fact $\{4\}$ will always be fixed at 0, because $\{4\}$ is always inhibited by $\{3\}$, so $\{5\}$ needs to be changed too.

4.2 SCC Filtering

Whether we want y to be the only reachable attractor, or merely to be one of potential several such attractors, the ordering from the previous part is the same, but the filtering will differ.

Theorem 2. *If a vertex u such as $x_u \neq y_u$ and u is not in a positive cycle, then modifying u's ancestors is sufficient to modify u.*
More generally, to switch from x to y, modifying only those strongly connected components that contain at least a positive cycle is sufficient.

Proof. Let u be a vertex such that $x_u \neq y_u$ and u does not lie in a positive cycle. If u is in a negative cycle, the incoming arc from the cycle is irrelevant : given that x and y are fixed points and that u has a distinct value in each, the negative cycle does not change u's value. Given that u is not in a positive cycle, u is not in a SCC (or not relevant if it is in a negative cycle). That means that none of the ancestors are descendants of u. Let z be the state where all of P_u (u's ancestors) have the same value that in y. By the remark from Sect.2, for all $v \in G[P_u]$, we have $f_v(z) = z_v = y_v$. So, either $f_u(z) = y_u$, and the theorem is proven, either $f_u(z) \neq y_u$, then, by Theor.1, u is in a positive cycle, contradiction. □

By recursion over the first part, modifying all the SCCs that contain positive cycles so their vertices have the same value as in y modifies all their children, and then all the children of their children, and so on, until the whole graph has the same values as y. □

Selecting the SCCs will differ with the two methods. It relies on the same base, searching the higher SCC that should have its values modified and that is not already selected. "Modified" means that all the values of the SCC are fixed to their values in y. The set of the selected SCCs is $\mathcal{S}$.

4.3 SCC Filtering for Existential Reachability

We consider the RDs for the BN reprogramming with Existential Reachability. We give an algorithm to identify different sets of SCCs for which the mutation in the initial fixed point ensure the reachability of the target fixed point. We will prove that the identified combination of SCCs is complete and minimal.

Basically, the algorithm reviews linearly the SCC slices according to $\prec$ and adds the minimal combinations of SCCs to $\mathcal{S}$ that are different in y and the fixed points reachable from $x_{[x_{\mathcal{S}}=y_{\mathcal{S}}]}$:

1. $\mathcal{S} := \emptyset$
2. For i ranging from 1 to c:
   - $T := \emptyset$
   - $\forall s \in P(C_i)$ such that s minimal
     $\exists z \in \{0,1\}^n$, $z_{C_i \setminus s} = y_{C_i \setminus s}$, $x_{[x_I = y_I | I \in \mathcal{S}]} \rightarrow^* z$, $T := T \cup s$.
   - $\mathcal{S} := \mathcal{S} \bar{\times} T$.

With $\bar{\times}$ being a product and union : for a set I of subsets $I_1, .., I_k$ and a set $J_1, .., J_l$, this product $\bar{\times}$ is defined by : $I \bar{\times} J = \{I_1 \cup J_1, .., I_1 \cup J_l, I_2 \cup J_1, ..., I_k \cup J_l\}$

Complexity : In the worst case, the above algorithm perform $c \times 2^l$ reachability checks (PSPACE-complete [5]), where l is the size of the largest slice.

Existence of a solution and proof of correctness : Forcing all SCCs of such problem that differ on x and y to have the same value as in y is one solution. In the worst case, that is what the algorithm will find. Since the algorithm tests reachability, and a solution exists, it will find one.

Example 5. We apply the algorithm on the BN of Fig.6 with $x = 00000$ and $y = 11011$.

1. $\mathcal{S} := \emptyset$
2. C_1: s minimal $\Leftrightarrow$ $s = \{1\}$
3. $\mathcal{S} := \mathcal{S} \bar{\times} \{1\} = \{\{1\}\}$
4. C_2: s minimal $\Leftrightarrow$ $s = \emptyset$ [3]
5. $\mathcal{S} := \mathcal{S} \bar{\times} \emptyset = \{\{1\}\}$.

We now prove the completeness of the algorithm and the minimality of the returned sets of SCCs (Theorem 3) and that any RDs in $ER(x, y)$ is spans only and necessarily in one of the set of SCCs identified by the algorithm (Theorem 4).

[3] with the path $10000 \rightarrow 10100 \rightarrow 10110 \rightarrow 11110 \rightarrow 11111$ (and the fixed point is the next step, $\rightarrow 11011$ but there is no need to go further than 11111.)

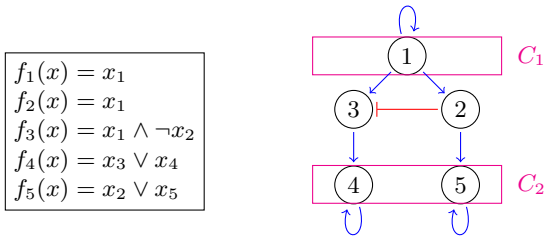

Fig. 6. BN of dimension 5 (left) with its interaction graph (right). Slices are enclosed in boxes. $C_1 = \{\{1\}\}$, $C_2 = \{\{4\}, \{5\}\}$.

Theorem 3. *$\mathcal{S}$ only contains minimal SCC sets, and $\mathcal{S}$ is complete.*

Proof. **Minimality :** Inside every slice, the SCCs are totally independant one another. Moreover, given the order exploiting, we can deduce that the sum of the minima on each slice is the minimum on the whole graph. □

Completeness : Let I be a minimal SCC set such as $x_{[x_J = y_J | J \in I]} \rightarrow^* y$, then, for every slice C_i, $I \cap C_i$ is minimal, since once all the SCCs in a slice can be changed to the way they are in y, we can always choose the path that allows this change. Hence $I \in \mathcal{S}$. □

Theorem 4. $\forall c \in ER(x, y)$, $\exists I \in \mathcal{S}$, $\forall u \in c$, $\exists scc \in I$, $u \in scc$.

Proof. Let c be a vertex set in $ER(x, y)$ and u one of the vertices. If $u \notin \mathcal{O}$, then c is not minimal, by Theorem 2. If for all $I \in \mathcal{S}$, u is in $o \in (\mathcal{O} \setminus I)$ then there exists a path such that changing o's ancestors makes o's change possible, and the ancestors need to be changed as well, by construction of I. So $c \setminus u$ would have the same effect, and c would not be minimal. If $u \notin o$, then there exists $I \in \mathcal{S}$ and $scc \in I$, such as $u \in scc$. □

4.4 SCC Filtering for Inevitable Reachability

We now give an algorithm to identify a set of SCCs for which the mutation in the initial fixed point is sufficient to ensure the *Inevitable Reachability* of the target fixed point.

The algorithm computes all reachable fixed points from x with the SCCs in $\mathcal{S}$ modified, and find the one, z, that has the lower SCC (in the ranking given by $\prec$) in which a vertex u is such that $z_u \neq y_u$. As we are looking for all reachable fixed points, this will always return the same SCC (even if the order is only partial), thus allowing the algorithm to be deterministic. We add this SCC to $\mathcal{S}$, and repeat until y is the only reachable fixed point.

1. $\mathcal{S} := \emptyset$
2. While $\exists z \in \mathrm{FP}(f)$, $z \neq y$, $x_{[x_I = y_I | I \in \mathcal{S}]} \rightarrow^* z$

– $\mathcal{S} := \mathcal{S} \cup \{\mathcal{O}_i\}$, with
$i = min_{a \in \{1,..,k\}}(a \mid \exists z \in \mathrm{FP}(f),\, z_{\mathcal{O}_a} \neq y_{\mathcal{O}_a},\, x_{[x_I = y_I | I \in \mathcal{S}]} \rightarrow^* z)$

If two (or more) SCCs A and B are such that they are differently ordered in two distinct orders, then A has no influence on B and neither has B on A. Then, the algorithm will select both SCCs if neither are impacted by the previous changes, so the order does not matter.

Existence of a solution and proof of correctness : A solution is to fix all the SCCs of the graph to their value in y. Since there exists a solution and the algorithm tests if y is the only reachable point, and follows the order given by $\prec$, it will end and find a solution.

Complexity : Computing all fixed points reachable is PSPACE-complete [4]. It is used k times (number of SCCs) in the worst case.

Example 6. We apply the algorithm on the BN of Fig.7. with $x = 00000$ and $y = 11011$. Starting from $\mathcal{S} := \emptyset$, the only reachable fixed point is (0)00(0)(0) (the SCCs from $\mathcal{O}$ are parenthesized). The smallest SCC o such as $x_{[x_\mathcal{S} = y_\mathcal{S}],o} \neq y_o$ is $\mathcal{O}_1$, so $\mathcal{S} := \emptyset \cup \{\mathcal{O}_1\} = \{\mathcal{O}_1\}$. The reachable fixed points from $x_{[x_I = y_I | I \in \mathcal{S}]} = 10000$ are now: (1)10(1)(1) and (1)10(0)(1). The smallest SCC o such that $x_{[x_\mathcal{S} = y_\mathcal{S}],o} \neq y_o$ is $\mathcal{O}_3$. We set $\mathcal{S} := \mathcal{S} \cup \mathcal{O}_3 = \{\mathcal{O}_1, \mathcal{O}_3\}$ and we obtain that the only reachable fixed point from $x_{[x_I = y_I | I \in \mathcal{S}]} = 10010$ is (1)10(1)(1) which is y. So the algorithm stops.

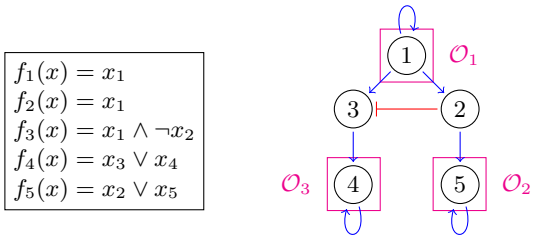

Fig. 7. BN of dimension 5 (left) with its interaction graph (right) on which the SCCs containing positive cycles ($\mathcal{O}$) are boxed.

Theorem 5. *$\mathcal{S}$ is minimal.*

Proof. If a set S_1 exists such that S_1 has a lower cardinal than $\mathcal{S}$ and modifying S_1 makes y the only reachable point, then we can reduce S_1 to a subset of $\mathcal{S}$. Let s be a SCC in $\mathcal{S} \setminus S_1$, thus there exists a fixed point z such that $z_s \neq y_s$ and by construction of $\mathcal{S}$, z is reachable from x modified by S_1. □

We remark that, contrary to the case of Existential Reachability, the RDs for Inevitable Reachability of the target fixed point are not necessarily in SCCs containing positive cycles. Indeed, in Ex.3, we showed that $IR_F(x, y)$ can refer to nodes that do not belong to $\mathcal{O}$ (such as the node 2 for the BN of Fig.3). But we can also remark that if a RD v is not in a SCC containing a positive cycle, then $x_v = y_v$.

Theorem 6. $\forall v \in IR(x, y)$, $x_v \neq y_v \Rightarrow \exists scc \in \mathcal{O}$, $v \in scc$

Proof. Let $v \in IR(x, y)$ such that for all $scc \in \mathcal{O}$, $v \notin scc$. By Theor.1, if v is such that $x_v \neq y_v$, then modifying the SCC in $\mathcal{O}$ is enough to modify v. But $v \in IR(x, y)$ and $IR(x, y)$ is minimal, thus $x_v = y_v$. □

5 Identifying Determinants within SCCs

We know that modifying all the SCCs selected is enough to switch from x to y, but to reduce the genes selected, we could try to modify only some of the vertices to achieve the same result. But, as dynamics are involved, there could be unwanted changes (or wanted and unpredicted changes, in the case where we want y to be reachable) in the descendants.

An idea could be to select the feedback vertex set of the SCC : by fixing the vertices from this set, we effectively destroy every circle, thus the only reachable state of the SCC is the one having the same values as y. This, however, does not solve the problem : in Ex.7, $\{1\}$ is the feedback vertex set, and we still have the same issue. Moreover, it miss some of the possible solutions (modifying $\{2\}$ or $\{3\}$ could work to change the whole SCC in Ex.7) or even dismiss the best solution (in Ex.7, changing $\{3\}$ makes y the only reachable fixed point and solves the issue).

Example 7. Illustration of the problem with dynamics.

$$
\begin{aligned}
f_1(x) &= \neg x_3 \wedge \neg x_2 \\
f_2(x) &= \neg x_1 \\
f_3(x) &= \neg x_1 \\
f_4(x) &= x_2 \wedge \neg x_1 \wedge \neg x_3 \\
f_5(x) &= x_4 \vee x_5
\end{aligned}
$$

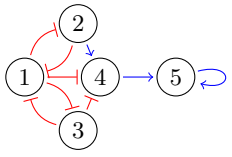

Fig. 8. BN of dimension 5 (left) and its interaction graph (right)

We decide that $x = 10000$ and $y = 01100$, and $01101 = z$, those are all fixed points. Let's suppose we want y to be the only fixed point reachable. The algorithm will see that if the whole first SCC is modified $\{1, 2, 3\}$, y is

the only reachable fixed point. It could pick $\{1\}$ to be modified, but instead of $00000 \to 00100 \to 01100$, we can have

$$00000 \to 01000 \to 01010 \to 01011 \to 01111 \to 01101$$

This leads to z being reachable by only modifying $\{1\}$, and so the algorithm would be wrong.

This leaves to two kinds of approaches : either a way to modify the SCC so that it does not impact its descendants can be found, either we need to select the vertices to be modified in the SCC as a intermediary step in the process, and redesign $\mathcal{S}$ as a list of vertices instead of a list of SCCs.

By exploiting the results of the preceding section, we show an algorithm to compute a set of RDs which guarantees the Inevitable Reachability of the target fixed point. The algorithm recursively picks a vertex u in the lowest SCC in the order given by $\prec$ in $\mathcal{O}$, and modify its associated function to become the constant value y_u. The interaction graph of the resulting Boolean network is a sub-graph of the initial interaction graph, where all the input edges of the node u have been removed. Hence, the SCC $\mathcal{O}_1$ is split in the new interaction graph. If necessary, another vertex can be picked in the lowest SCC in the new interaction graph:

RecursiveAlgorithm(f, rd) :

- If $\exists z \in \mathrm{FP}(f)$, $x_{[x_{rd}=y_{rd}]} \to^* z$ then :
  - $res = \emptyset$
  - $i = min_{a \in \{1,..,k\}}(a \mid \exists z \in \mathrm{FP}(f),\, z_{\mathcal{O}_a} \neq y_{\mathcal{O}_a},\, x_{[x_I=y_I | I \in \mathcal{S}]} \to^* z)$
  - For all $u \in \mathcal{O}_i$:
    - $g := f$ with $g_u := y_u$
    - $res := res \cup$ RecursiveAlgorithm(g, $rd \bar{\times} \{u\}$)
  - return res
- else :
  - return rd

Remark that the algorithm always find at least one solution: if the target fixed point is not the only reachable fixed point, then there is at least one positive cycle (and hence a SCC) which has a different state (and hence will be selected by our algorithm).

Example 8. Applied to the BN of Fig.8 with $x = 10000$ and $y = 01100$, the above algorithm returns, for instance, the RD $\{2, 5\}$: indeed, $\{2\}$ belongs to $\mathcal{O}_1$. When fixing $f_2 = 1$, the new interaction graph has two SCCs with positive cycles: $\{1, 3\}$ and $\{5\}$. From the state 11000, two fixed points are reachable: 01100, 01101. Hence, because the SCC $\{1, 3\}$ has the same values than in y in those two fixed points, the next vertex in picked in the SCC $\{5\}$. Finally, from the state 11001, y is the only reachable fixed point.

6 Discussion

This paper provides the first formal characterization of the Reprogramming Determinants (RDs) for switching from one fixed point to another in the scope of the asynchronous dynamics of Boolean networks.

In the case of reprogramming with existential reachability of the target fixed point, we prove that all the possible minimal RDs modify nodes in particular combinations of SCCs of the interaction graph of the Boolean network. We give an algorithm to determine exactly those combinations of set of nodes. Our characterizations rely on the verification of reachability properties.

In the case of reprogramming with inevitable reachability of the target fixed point, we show that the RDs are not necessarily in SCCs. However, we provide an algorithm which identifies RDs that guarantee the inevitable reachability by picking nodes in appropriate SCCs. The algorithm relies on the enumeration of reachable fixed points.

One of the main limitation of our algorithms is the numerous reachability checks it needs to perform. Future work will consider methods and data structures for factorizing the exploration of the Boolean network dynamics.

The present work considered only permanent mutations: when a node is mutated, it is assumed it keeps its mutated value forever (its local Boolean function becomes a constant function). Considering temporary mutations, i.e., where the local Boolean function of mutated nodes is restored after some time, is a challenging research direction: one should determine the ordering and the duration of mutations, and the set of candidate mutations is *a priori* no longer restricted to connected components, as it is the case for permanent mutations.

References

1. Wassim Abou-Jaoudé, Pedro T. Monteiro, Aurélien Naldi, Maximilien Grandclaudon, Vassili Soumelis, Claudine Chaouiya, and Denis Thieffry. Model checking to assess t-helper cell plasticity. *Frontiers in Bioengineering and Biotechnology*, 2, 2015.
2. Julio Aracena. Maximum number of fixed points in regulatory boolean networks. *Bulletin of Mathematical Biology*, 70(5):1398–1409, feb 2008.
3. Rui Chang, Robert Shoemaker, and Wei Wang. Systematic search for recipes to generate induced pluripotent stem cells. *PLoS Computational Biology*, 7(12):e1002300, Dec 2011.
4. Thomas Chatain, Stefan Haar, Loïg Jezequel, Loïc Paulevé, and Stefan Schwoon. Characterization of reachable attractors using Petri net unfoldings. In Pedro Mendes, Joseph Dada, and Kieran Smallbone, editors, *Computational Methods in Systems Biology*, volume 8859 of *Lecture Notes in Computer Science*, pages 129–142. Springer Berlin Heidelberg, 2014.
5. Allan Cheng, Javier Esparza, and Jens Palsberg. Complexity results for 1-safe nets. *Theor. Comput. Sci.*, 147(1&2):117–136, 1995.
6. Isaac Crespo, Thanneer M Perumal, Wiktor Jurkowski, and Antonio del Sol. Detecting cellular reprogramming determinants by differential stability analysis of gene regulatory networks. *BMC Systems Biology*, 7(1):140, 2013.

7. Antonio del Sol and Noel J. Buckley. Concise review: A population shift view of cellular reprogramming. *STEM CELLS*, 32(6):1367–1372, 2014.
8. Zuguang Gao, Xudong Chen, and Tamer Başar. On the stability of conjunctive boolean networks. March 2016.
9. Thomas Graf and Tariq Enver. Forcing cells to change lineages. *Nature*, 462(7273):587–594, dec 2009.
10. Junghyun Jo, Sohyun Hwang, Hyung Joon Kim, Soomin Hong, Jeoung Eun Lee, Sung-Geum Lee, Ahmi Baek, Heonjong Han, Jin Il Lee, Insuk Lee, and et al. An integrated systems biology approach identifies positive cofactor 4 as a factor that increases reprogramming efficiency. *Nucleic Acids Research*, 44(3):1203–1215, Jan 2016.
11. Loïc Paulevé. Goal-oriented reduction of automata networks. In *CMSB 2016 - 14th conference on Computational Methods for Systems Biology*, 2016. accepted.
12. Owen J L Rackham, Jaber Firas, Hai Fang, Matt E Oates, Melissa L Holmes, Anja S Knaupp, Harukazu Suzuki, Christian M Nefzger, Carsten O Daub, Jay W Shin, Enrico Petretto, Alistair R R Forrest, Yoshihide Hayashizaki, Jose M Polo, and Julian Gough. A predictive computational framework for direct reprogramming between human cell types. *Nature Genetics*, 48(3):331–335, jan 2016.
13. Adrien Richard. Positive circuits and maximal number of fixed points in discrete dynamical systems. *Discrete Applied Mathematics*, 157(15):3281 – 3288, 2009.
14. Adrien Richard. Negative circuits and sustained oscillations in asynchronous automata networks. *Advances in Applied Mathematics*, 44(4):378 – 392, 2010.
15. Kazutoshi Takahashi and Shinya Yamanaka. A decade of transcription factor-mediated reprogramming to pluripotency. *Nat Rev Mol Cell Biol*, 17(3):183–193, Feb 2016.
16. René Thomas. Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, 42(3):563 – 585, 1973.
17. Élisabeth Remy, Paul Ruet, and Denis Thieffry. Graphic requirements for multistability and attractive cycles in a boolean dynamical framework. *Advances in Applied Mathematics*, 41(3):335 – 350, 2008.