Computing boolean network attractors for target identification

Arnaud Poret[*]

Contents

Abstract

Target identification aims at identifying, in pathologically disturbed biological systems, biomolecules whose function should be therapeutically modified in order to alleviate, or ultimately cure, the physiological consequences of the corresponding pathology. In an attempt to improve the efforts made in this area thanks to *in silico* tools, an algorithm based on the computation of boolean network attractors is proposed. It assumes that the attractors of a dynamical system, such as a boolean network, correspond to the phenotypes produced by the modeled biological network. Under this assumption, the algorithm identifies target combinations together with the therapeutic modifications to be applied to them, here

[*]Copyright (c) 2013-2014, Arnaud Poret. All rights reserved. arnaud.poret@gmail.com

called bullets, which allow pathologically disturbed biological networks to avoid attractors responsible for pathological phenotypes.

The algorithm was tested on a published boolean model of the mammalian cell cycle where the retinoblastoma protein was constitutively inactivated as seen in diseases such as cancer. It returned four effective bullets able to remove the attractors responsible for pathological phenotypes, that is the attractors not reached by the non-pathologically disturbed biological network.

Under the assumption linking dynamical system attractors and biological network phenotypes, the results showed that the algorithm succeeds in performing the proposed *in silico* target identification. However, as with any *in silico* evidence, it should be considered as such: there is a bridge to cross between theory and practice. Nevertheless, it is expected that the algorithm will take its place in, and contribute to, work done in target identification.

1 Background

Target discovery aims at identifying new effective drugs against diseases. This process can be segmented into three steps [1]: i) disease model provision, where experimental models of the disease are developed, ii) target identification, where new therapeutic targets are proposed and iii) target validation, where the proposed therapeutic targets are assessed. The present work focuses on the second step of target discovery: target identification.

Given an organism suffering from a disease, it aims at finding where to act among its multitude of biomolecules in order to alleviate, or ultimately to cure, the physiological consequences of the disease, hence improving the life quality or quantity of the ill organism [2]. These biomolecules on which perturbations should be applied are called targets and are targeted by drugs [3]. This raises two questions: which target should be therapeutically perturbed and what type of perturbation should be applied to it. These two questions can be unified by stating that the perturbation aims at modifying the biological function of the target thanks to its interaction with a drug. Broadly, the functional modification of a target by a drug can be either activating or inactivating, regardless the way the drug achieves it.

Basically, given an organism and a disease, one solution is to test all, or at least a large number of, its biomolecules for activation or inactivation and knowing that it is possibly more effective to target more than one biomolecule [4], the number of possibilities is consequently huge [5]. This rather brute force method can be refined with knowledge about the disease pathophysiology by identifying some potential targets based on the role their biological function plays in it [6]. Even with this knowledge, experimentally assessing *in vitro* or *in vivo* the selected potential targets is far from being straightforward. Indeed, such experimentations are costly in time and resources and exhibit a high risk of failure [7]. Fortunately, *in silico* experimentations appear as valuable tools in improving the efficiency of therapeutic researches [8, 9]. Indeed, *in silico* experimentations are less costly in time and resources than the traditional *in vitro* and *in vivo* experimentations. However, the stumbling block of *in silico* experimentations is that they are built from the available knowledge about the pathophysiology of interest: not all is known about everything. Nevertheless, an impressive and ever increasing amount of biological knowledge is already available through the scientific literature, databases and knowledge bases [10, 11, 12, 13, 14]. Consequently, this increasing body of knowledge becomes harder and harder to integrate by humans alone because of the complexity of biological systems [15]: this is where computational tools come into play [16]. This interplay between the traditional and the computational biology is synergistic rather than competing [17]. Indeed, since *in vitro* and *in vivo* experimentations are rather factual, they are a relatively trustworthy source of knowledge. Once these factual pieces of knowledge are obtained, computational tools can help to integrate them in order to infer new knowledge. Afterwards, this computationally obtained knowledge can be used to direct further *in vitro* or *in vivo* experimentations, hence mutually potentiating the whole.

The goal of the present work is to propose a computational method implemented in an algorithm for target identification based on the computation of boolean network attractors [18]. It is based on the assumption that the attractors of a boolean network correspond to the phenotypes produced by the modeled biological network, an assumption successfully applied in several computations [19, 20, 21, 22, 23, 24]. Indeed, assuming that a phenotype is an observable and hence a relatively stable state of a biological system and assuming that the state of a biological system is due to the dynamic of the underlying biological network, a phenotype is likely to correspond to an attractor. This assumption can be stated for any dynamical model of a biological network but, in this work, only boolean networks are considered. The reason is that, in their most basic form, boolean networks do not require parameter values and that parameter values are not straightforward to obtain in biology [25], particularly at the sub-cellular scale, the scale where drugs interact with their targets. Moreover, since synchronous boolean networks are easier to compute than asynchronous ones [26] and given that this work focuses essentially on a computational methodology, only synchronous boolean networks are considered. This does not exclude the possibility, at a later stage, to extend the algorithm for both synchronous and asynchronous boolean networks.

For a biological network involved in a disease pathophysiology, two possible variants are considered: the physiological variant, bore by healthy organisms, which produces physiological phenotypes and the pathological variant, bore by ill organisms, which produces pathological phenotypes or which fails to produce physiological ones. A physiological phenotype does not impair the life quantity or quality of the organism which exhibits it while a pathological phenotype does. It should be noted that the loss of a physiological phenotype is also a pathological condition. The physiological and pathological variants differ in that the second one results from the occurrence of some structural modifications in the network known to be responsible for disorders. Hence, with a pathological variant, there are two non-exclusives pathological scenarios: pathological phenotypes are gained or physiological phenotypes are lost.

The primary goal of the algorithm is to identify, in the pathological variant, target combinations together with the kind of perturbation to apply to them, here called bullets, which make it unable at producing pathological phenotypes. The secondary goal is to classify the obtained bullets according to their ability at making the pathological variant producing the lost physiological phenotypes, if any.

2 Methods

2.1 Preliminaries

This section briefly introduces biological networks [27] and boolean networks [28].

2.1.1 Biological networks

A network can be seen as a graph $G = (V, E)$ where $V = \{v_1, \dots, v_n\}$ is the set of size n containing exactly all the nodes v_i of the network and where $E = \{(v_{i,1}, v_{j,1}), \dots, (v_{i,m}, v_{j,m})\} \subseteq V^2$ is the set of size m containing exactly all the edges (v_i, v_j) of the network. In practice, nodes represent things and edges represent binary relations $R \subseteq V^2$ involving these things: $v_i \; R \; v_j$. In biological networks, nodes should represent biological entities and edges should represent biological relations. For example, in gene regulatory networks, nodes represent gene products and edges represent gene expression modulation.

2.1.2 Boolean networks

A boolean network is a network where nodes are boolean variables x_i and where edges (x_i, x_j) represent the binary *is input of* relation: x_i *is input of* x_j. Each x_i has $b_i \in [\![0, n]\!]$ inputs $x_{i,1}, \dots, x_{i,b_i}$ and the variables which are not input of x_i have no direct influence on it. In the case where $b_i = 0$, x_i is a parameter: it does not depend on other variables. At each iteration $k \in [\![k_0, k_{end}]\!]$ of the computation, the value $x_i(k) \in \{0, 1\}$ of each x_i is updated to the value $x_i(k+1)$ thanks to a boolean function f_i and to the values $x_{i,1}(k), \dots, x_{i,b_i}(k)$ of its inputs:

```
1 for $k \in [\![k_0, k_{end} - 1]\!]$ do
2     $x_1(k+1) = f_1(x_{1,1}(k), \dots, x_{1,b_1}(k))$
3     ...
4     $x_n(k+1) = f_n(x_{n,1}(k), \dots, x_{n,b_n}(k))$
5 end for
```

which can be written in a more concise form:

```
1 for $k \in [\![k_0, k_{end} - 1]\!]$ do
2     $\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k))$
3 end for
```

where $\boldsymbol{f} = (f_1, \dots, f_n)$ is the boolean transition function and $\boldsymbol{x} = (x_1, \dots, x_n)$ is the state vector. The value $\boldsymbol{x}(k) = (x_1(k), \dots, x_n(k)) \in \{0, 1\}^n$ of $\boldsymbol{x}$ at k belongs to the state space $S = \{0, 1\}^n$ which is the set of size 2^n containing exactly all the possible states. If the value of all the x_i is updated simultaneously at each k then the network is synchronous, otherwise it is asynchronous. With synchronous boolean networks, at each k, $\boldsymbol{x}(k)$ has a unique possible successor $\boldsymbol{x}(k+1)$: synchronous boolean networks are deterministic.

In the particular case where $k = k_0$, $\boldsymbol{x}(k_0) = \boldsymbol{x}_0$ is the initial state and, in deterministic dynamical systems such as synchronous boolean networks, determines entirely the trajectory $w = (\boldsymbol{x}(k_0), \dots, \boldsymbol{x}(k_{end}))$. w is a sequence of length k_{end}[1] and results from the iterative computation of $\boldsymbol{x}(k)$ from k_0 up to k_{end}.

[1] It is assumed that $k_0 = 1$.

This iterative computation can be seen as the discretization of a time interval: boolean networks are discrete dynamical systems as they simulate, discretely, the time course of the state vector. Furthermore, while boolean networks are networks on their own, in practice they can be used to model other networks such as biological networks: boolean networks are discrete dynamical systems which can be used to model the dynamic of biological networks.

The set $A = \{a_1, \dots, a_p\}$ of size p containing exactly all the attractors a_i is the attractor set. Due to the determinism of synchronous boolean networks, all the attractors are cycles. A cycle is a sequence $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q)$ of length q such that $\forall j \in [\![1, q]\!]$, $\boldsymbol{x}_{j+1} = \boldsymbol{f}(\boldsymbol{x}_j)$ and $\boldsymbol{x}_{q+1} = \boldsymbol{x}_1$: once the system reach a state $\boldsymbol{x}_j$ belonging to a cycle, it will successively visit its states $\boldsymbol{x}_{j+1}, \dots, \boldsymbol{x}_q, \boldsymbol{x}_1, \dots, \boldsymbol{x}_j$ for infinity. In the particular case where $q = 1$, the cycle is a point attractor. The set $B_i \subseteq S$ containing exactly all the $\boldsymbol{x} \in S$ from which a_i can be reached is its basin of attraction. With deterministic dynamical systems such as synchronous boolean networks, the family of sets $(B_1, \dots, B_p)$ is a partition of S.

2.2 Definitions

Some particular concepts used in this work should be formally defined. Given a synchronous boolean network modeling the dynamic of a biological network bore by an organism:

- *physiological phenotype*: a phenotype which does not impair the quantity or quality of life of the organism which exhibits it.
- *pathological phenotype*: a phenotype which impairs the quantity or quality of life of the organism which exhibits it.
- *variant (of a biological network)*: a biological network obtained from the original one where some modifications have occurred in its structure.
- *physiological variant*: a variant of the biological network which produces only physiological phenotypes.
- *pathological variant*: a variant of the biological network which produces at least one pathological phenotype.
- *physiological attractor set*: the attractor set A_{physio} of the physiological variant.
- *pathological attractor set*: the attractor set A_{patho} of the pathological variant.
- *physiological boolean transition function*: the boolean transition function $\boldsymbol{f}_{physio}$ of the physiological variant.
- *pathological boolean transition function*: the boolean transition function $\boldsymbol{f}_{patho}$ of the pathological variant.
- *run*: an iterative computation of the state vector value $\boldsymbol{x}(k)$ starting from an initial state $\boldsymbol{x}(k_0) = \boldsymbol{x}_0$ until an attractor a_i is reached. It returns the trajectory $w = (\boldsymbol{x}(k_0), \dots, \boldsymbol{x}(k_{end}))$ where k_{end} depends on when a_i is reached and hence on $\boldsymbol{x}_0$.

- *physiological attractor*: an attractor a_i such that $a_i \in A_{physio}$
- *pathological attractor*: an attractor a_i such that $a_i \notin A_{physio}$
- *modality*: the nature of the functional perturbation $moda_i$ applied to an entity $v_j \in V$ of the network, either activating ($moda_i = 1$) or inactivating ($moda_i = 0$): at each iteration k, $moda_i$ overwrites $f_j(\boldsymbol{x}(k))$ and hence $x_j(k+1) = moda_i$.
- *target*: an entity $v_j \in V$ of the network on which a modality $moda_i$ is applied.
- *bullet*: a couple (c_{targ}, c_{moda}) where $c_{targ} = (targ_1, \ldots, targ_r)$ is a combination without repetition of targets $targ_i$ and where $c_{moda} = (moda_1, \ldots, moda_r)$ is an arrangement with repetition of modalities $moda_i$, $r \in [\![1, n]\!]$ being the number of targets in the bullet. $moda_i$ is intended to be applied to $targ_i$.
- *therapeutic bullet*: a bullet which makes $A_{patho} \subseteq A_{physio}$.
- *silver bullet*: a therapeutic bullet which makes $A_{patho} \subsetneq A_{physio}$.
- *golden bullet*: a therapeutic bullet which makes $A_{patho} = A_{physio}$.

The assumed link between phenotypes and attractors is the reason why attractors are qualified as either physiological or pathological according to the phenotype they produce and the reason why target identification aims at manipulating the attractor set of the pathological variant.

2.3 Steps of the algorithm

The algorithm has two goals: i) finding therapeutic bullets and ii) classifying them as either golden or silver. A therapeutic bullet is intended to make the pathological variant unable to reach pathological attractors, that is to make $A_{patho} \subseteq A_{physio}$. If such a bullet is applied to the pathological variant, the organism bearing it no longer exhibits the associated pathological phenotypes: this is the primary goal. However, a therapeutic bullet does not necessarily preserve or restore all the physiological attractors. If a therapeutic bullet preserves or restores all the physiological attractors, that is if $A_{patho} = A_{physio}$, it is a golden one but if $A_{patho} \subsetneq A_{physio}$ it is a silver one: this classification is the secondary goal.

To do this, given a physiological and a pathological variant, that is $\boldsymbol{f}_{physio}$ and $\boldsymbol{f}_{patho}$, the algorithm follows five steps:

1. with $\boldsymbol{f}_{physio}$ it computes the control attractor set A_{physio}
2. it randomly generates bullets and for each of them it performs the three following steps
3. with $\boldsymbol{f}_{patho}$ under the effect of the bullet it computes the variant attractor set A_{patho}
4. it assesses the therapeutic potential of the bullet by comparing A_{physio} and A_{patho} to detect pathological attractors

5. if the bullet is a therapeutic one, it classifies it as either golden or silver by comparing A_{physio} and A_{patho} for equality

which can be written:

```
with $\boldsymbol{f}_{physio}$ compute $A_{physio}$
generate $bullet\_set$
**for** $bullet \in bullet\_set$ **do**
    with $\boldsymbol{f}_{patho}$ plus $bullet$ compute $A_{patho}$
    **if** $A_{patho} \subseteq A_{physio}$ **then**
        $bullet$ is therapeutic
        **if** $A_{patho} = A_{physio}$ **then**
            $bullet$ is golden
        **else**
            $bullet$ is silver
        **end if**
    **end if**
**end for**
```

2.3.1 Step 1: computing A_{physio}

First of all, A_{physio} has to be computed since it is used as the control and hence determines what is pathological and what is not. To this end, runs are performed with $\boldsymbol{f}_{physio}$ and the reached a_i are stored in A_{physio}. However, $\boldsymbol{x}_0 \in S$ and $size(S)$ increases exponentially with n. Even for a reasonable n, $size(S)$ explodes: more than one million of possible $\boldsymbol{x}_0$ for $n = 20$. One solution ensuring that all the possible a_i are reached is to start a run from each of the possible $\boldsymbol{x}_0$, that is from each of the $\boldsymbol{x} \in S$. In practice this is not feasible for an arbitrary n because the required computational resources can be too demanding. For example, assuming that a run requires one millisecond of computation and assuming that $n = 50$, performing a run starting from each of the 2^{50} $\boldsymbol{x} \in S$ requires nearly 36 thousand years, definitively not a practical duration. However, given that with deterministic dynamical systems such as synchronous boolean networks $(B_1, \ldots, B_p)$ is a partition of S, a solution is to randomly select a subset $D \subseteq S$ of a given size containing the $\boldsymbol{x}_0$ to start from. In order to avoid bias in the random selection, a uniform distribution should be used. The stumbling block of this solution is that it does not ensure that at least one $\boldsymbol{x}_0$ per B_i is selected and then it does not ensure that all the a_i are reached. This stumbling block holds only if $size(D) < size(S)$.

Again given that synchronous boolean networks are deterministic, if a run visits a state which has already been visited during a previous run, its destination, that is the attractor it will reach, has already been found. Hence, the run can be immediately stopped and the algorithm can directly jump to the next one. To implement this, the previous trajectories are stored in a set H, the history, and at each k the algorithm checks if $\exists w \in H : \boldsymbol{x}(k) \in w$. If this check is positive then the algorithm jumps to the next run.

Identifying the attractors is relatively easy with deterministic dynamical systems such as synchronous boolean networks. Indeed, in this case, attractors are cycles. Therefore, to identify a cycle, the algorithm checks at each k if $\boldsymbol{x}(k+1)$ has already been visited during the current run, that is if $\exists k' \in [\![1, k]\!] : \boldsymbol{x}(k+1) = \boldsymbol{x}(k')$. If this check is positive then $a_i = (\boldsymbol{x}(k'), \ldots, \boldsymbol{x}(k))$.

This step can be written:

```
input $size(D)$
1  $size(D) = min(size(D), 2^n)$
2  generate $D \subseteq S$
3  $H = \{\}$
4  $A_{physio} = \{\}$
5  for $x_0 \in D$ do
6      $k = 1$
7      $\boldsymbol{x}(1) = x_0$
8      while 1 do
9          if $\exists w \in H : \boldsymbol{x}(k) \in w$ then
10             break
11         end if
12         $\boldsymbol{x}(k+1) = \boldsymbol{f}_{physio}(\boldsymbol{x}(k))$
13         if $\exists k' \in [\![1, k]\!] : \boldsymbol{x}(k+1) = \boldsymbol{x}(k')$ then
14             $A_{physio} = A_{physio} \cup \{(\boldsymbol{x}(k'), \ldots, \boldsymbol{x}(k))\}$
15             break
16         end if
17         $k = k + 1$
18     end while
19     $H = H \cup \{(\boldsymbol{x}(1), \ldots, \boldsymbol{x}(k))\}$
20 end for
output $A_{physio}$
21 do step 2
```

Line 2 catches the mistake $size(D) > size(S)$. It should be noted that more sophisticated algorithms intended to find boolean network attractors are already available, such as the ones proposed by Zheng *et al* [29].

2.3.2 Step 2: generating bullets

The bullets are the candidate perturbations to apply to the pathological variant to make it unable to reach pathological attractors and hence to make it unable to produce pathological phenotypes. Generating a bullet is to choose some $targ_i \in V$ and to choose for each of them a $moda_i \in \{0, 1\}$. In this work there is no time sequencing in the target engagement nor in the modality choice. This means that, given a bullet and during a run, all the $targ_i$ are engaged simultaneously and constantly and that the $moda_i$ do not change during the run. Hence, in this setting, choosing more than once the same $targ_i$ is senseless while it is possible to choose the same $moda_i$ for more than one $targ_i$. Therefore, a bullet is a combination c_{targ} without repetition of $targ_i$ together with an arrangement c_{moda} with repetition of $moda_i$. If bullets containing r targets have to be generated then there are $n!/(r! \cdot (n-r)!)$ possible c_{targ} and for each of them there are 2^r possible c_{moda}. This raises the same computational problem as with the state space size explosion since there are $(n! \cdot 2^r)/(r! \cdot (n-r)!)$ possible bullets. For example, assuming that $n = 50$ and that $r = 3$, there are more than 150 thousand possible bullets. Knowing that the algorithm, as explained below, computes one attractor set per bullet, the computation time becomes practically unfeasible. To overcome this, the algorithm asks for r as an interval $[\![r_{min}, r_{max}]\!]$, asks for a maximum number max_{targ} of c_{targ} to generate and a maximum number max_{moda} of c_{moda} to test for each c_{targ}. Then the algorithm generates a set C_{targ} of c_{targ} with $size(C_{targ}) \leq max_{targ}$ by randomly selecting, along a uniform distribution and without repetition, entities in the network. In the same way, the algorithm generates a set C_{moda} of c_{moda} with $size(C_{moda}) \leq max_{moda}$ by randomly choosing, along a uniform distribution and with repetition, modalities as either activating ($= 1$) or inactivating ($= 0$). The result is the bullets: per $r \in [\![r_{min}, r_{max}]\!]$, a C_{targ} together with a C_{moda}. As with the state space size explosion, the stumbling block of this method is that it does not ensure that all the possible c_{targ} together with all the possible c_{moda} are tested. This stumbling block holds only if $max_{targ} < n!/(r! \cdot (n-r)!)$ or $max_{moda} < 2^r$.

This step can be written:

```
**input** $r_{min}, r_{max}, max_{targ}, max_{moda}$
 1 $r_{max} = min(r_{max}, n)$
 2 $golden\_set = \{\}$
 3 $silver\_set = \{\}$
 4 **for** $r \in [\![r_{min}, r_{max}]\!]$ **do**
 5     $max^r_{targ} = min(max_{targ}, n!/(r! \cdot (n-r)!))$
 6     $max^r_{moda} = min(max_{moda}, 2^r)$
 7     $C_{targ} = \{\}$
 8     $C_{moda} = \{\}$
 9     **while** $size(C_{targ}) < max^r_{targ}$ **do**
10         generate $c_{targ} \notin C_{targ}$
11         $C_{targ} = C_{targ} \cup \{c_{targ}\}$
12     **end while**
13     **while** $size(C_{moda}) < max^r_{moda}$ **do**
14         generate $c_{moda} \notin C_{moda}$
15         $C_{moda} = C_{moda} \cup \{c_{moda}\}$
16     **end while**
17     do steps 3 to 5
18 **end for**
**output** $golden\_set$
**output** $silver\_set$
```

Line 2 catches the mistake $r > n$. Lines 3–4 create the sets in which the therapeutic bullets which will be found during the step 4 will be classified as either golden or silver during the step 5. Lines 6–7 catch the mistake where max_{targ} or max_{moda} is greater than its maximum, which depends on r hence the creation of max^r_{targ} or max^r_{moda} to preserve the original supplied value. Lines 11,15 ensure that only novel c_{targ} and c_{moda} are generated.

2.3.3 Step 3: computing A_{patho}

Having the control attractor set A_{physio} and a bullet $(c_{targ}, c_{moda}) \in C_{targ} \times C_{moda}$, the algorithm computes the variant attractor set A_{patho} under the effect of the bullet by almost the same way A_{physio} was computed during the step 1. However, in this step, $\boldsymbol{f}_{patho}$ is used instead of $\boldsymbol{f}_{physio}$ and the bullet is applied: at each k, $f_j(\boldsymbol{x}(k))$ is overwritten by $moda_i \in c_{moda}$, that is $x_j(k+1) = moda_i$, provided that $v_j = targ_i \in c_{targ}$.

In order to apply all the generated bullets, the algorithm uses two nested for loops: for each $c_{targ} \in C_{targ}$ it uses successively all the $c_{moda} \in C_{moda}$: C_{moda} contains all the c_{moda} to test for each $c_{targ} \in C_{targ}$. For each of these couples

(c_{targ}, c_{moda}) the algorithm computes the corresponding A_{patho} and does the steps 4 and 5.

This step can be written:

```
 1 for $c_{targ} \in C_{targ}$ do
 2     for $c_{moda} \in C_{moda}$ do
 3         $H = \{\}$
 4         $A_{patho} = \{\}$
 5         for $x_0 \in D$ do
 6             $k = 1$
 7             $\boldsymbol{x}(1) = x_0$
 8             while 1 do
 9                 if $\exists w \in H : \boldsymbol{x}(k) \in w$ then
10                     break
11                 end if
12                 $\boldsymbol{x}(k+1) = \boldsymbol{f}_{patho}(\boldsymbol{x}(k))$
13                 for $targ_i \in c_{targ}$ do
14                     for $v_j \in V$ do
15                         if $v_j = targ_i$ then
16                             $x_j(k+1) = moda_i$
17                         end if
18                     end for
19                 end for
20                 if $\exists k' \in [\![1, k]\!] : \boldsymbol{x}(k+1) = \boldsymbol{x}(k')$ then
21                     $A_{patho} = A_{patho} \cup \{(\boldsymbol{x}(k'), \ldots, \boldsymbol{x}(k))\}$
22                     break
23                 end if
24                 $k = k + 1$
25             end while
26             $H = H \cup \{(\boldsymbol{x}(1), \ldots, \boldsymbol{x}(k))\}$
27         end for
28         do step 4 and 5
29     end for
30 end for
```

Lines 13–19 are where the bullet is applied.

2.3.4 Step 4: identifying therapeutic bullets

To identify therapeutic bullets among the generated ones, for each tested bullet in the step 3 and once the corresponding A_{patho} is obtained, the algorithm compares it with A_{physio} to check if $A_{patho} \subseteq A_{physio}$. This check ensures that, under the effect of the bullet, all the pathological attractors were removed and ensures that if new attractors have appeared they are physiological ones. If this check is positive then the bullet is a therapeutic one and the algorithm pursues with the step 5.

This step can be written:

```
1 if $A_{patho} \subseteq A_{physio}$ then
2     do step 5
3 end if
```

2.3.5 Step 5: assessing therapeutic bullets

The therapeutic bullets are qualified as either silver or golden according to their ability at making the pathological variant able to reach the physiological attractors. Indeed, all therapeutic bullets, either golden or silver, are primarily intended to remove all the pathological attractors without creating new ones, that is to make $A_{patho} \subseteq A_{physio}$. However, this does not imply that all the therapeutic bullets preserve or restore the physiological attractors. This is why they are classified as either golden or silver. A golden bullet preserves or restores all the physiological attractors: $A_{patho} = A_{physio}$ while a silver bullet does not: $A_{patho} \subsetneq A_{physio}$.

Hence, in the setting of synchronous boolean modeling, golden bullets are perfect therapies while silver bullets are not. However, since precious things are rare and since gold is rarer than silver, finding a golden bullet is less likely than finding a silver one. Indeed, given that more constraints are required for a therapeutic bullet to be a golden one, it is more likely that the identified therapeutic bullets are silver ones.

Practically, again in the setting of synchronous boolean modeling, an organism which bears a pathological variant treated by a therapeutic bullet no longer exhibits the associated pathological phenotypes. Moreover, if the therapeutic bullet is a golden one, the organism has the same phenotype than its healthy counterpart. However, if the therapeutic bullet is a silver one, the organism fails to exhibit at least one of the physiological phenotypes. With a silver bullet this is a matter of choice: what is the less detrimental for the ill organism between a silver bullet and no therapeutic bullet.

This step can be written:

```
1 if $A_{patho} = A_{physio}$ then
2     $golden\_set = golden\_set \cup \{(c_{targ}, c_{moda})\}$
3 else
4     $silver\_set = silver\_set \cup \{(c_{targ}, c_{moda})\}$
5 end if
```

2.4 The algorithm in one block

The algorithm was described step by step for the sake of clarity. Below is the algorithm in one block.

```
input $size(D)$
 1 $size(D) = min(size(D), 2^n)$
 2 generate $D \subseteq S$
 3 $H = \{\}$
 4 $A_{physio} = \{\}$
 5 for $x_0 \in D$ do
 6     $k = 1$
 7     $\boldsymbol{x}(1) = x_0$
 8     while $1$ do
 9         if $\exists w \in H : \boldsymbol{x}(k) \in w$ then
10             break
11         end if
12         $\boldsymbol{x}(k+1) = \boldsymbol{f}_{physio}(\boldsymbol{x}(k))$
13         if $\exists k' \in [\![1, k]\!] : \boldsymbol{x}(k+1) = \boldsymbol{x}(k')$ then
```

```
            $A_{physio} = A_{physio} \cup \{(\boldsymbol{x}(k'), \ldots, \boldsymbol{x}(k))\}$
            **break**
        **end if**
        $k = k + 1$
    **end while**
    $H = H \cup \{(\boldsymbol{x}(1), \ldots, \boldsymbol{x}(k))\}$
**end for**
**output** $A_{physio}$
**input** $r_{min}, r_{max}, max_{targ}, max_{moda}$
$r_{max} = min(r_{max}, n)$
$golden\_set = \{\}$
$silver\_set = \{\}$
**for** $r \in [\![r_{min}, r_{max}]\!]$ **do**
    $max^r_{targ} = min(max_{targ}, n!/(r! \cdot (n-r)!))$
    $max^r_{moda} = min(max_{moda}, 2^r)$
    $C_{targ} = \{\}$
    $C_{moda} = \{\}$
    **while** $size(C_{targ}) < max^r_{targ}$ **do**
        generate $c_{targ} \notin C_{targ}$
        $C_{targ} = C_{targ} \cup \{c_{targ}\}$
    **end while**
    **while** $size(C_{moda}) < max^r_{moda}$ **do**
        generate $c_{moda} \notin C_{moda}$
        $C_{moda} = C_{moda} \cup \{c_{moda}\}$
    **end while**
    **for** $c_{targ} \in C_{targ}$ **do**
        **for** $c_{moda} \in C_{moda}$ **do**
            $H = \{\}$
            $A_{patho} = \{\}$
            **for** $x_0 \in D$ **do**
                $k = 1$
                $\boldsymbol{x}(1) = x_0$
                **while** $1$ **do**
                    **if** $\exists w \in H : \boldsymbol{x}(k) \in w$ **then**
                        **break**
                    **end if**
                    $\boldsymbol{x}(k+1) = \boldsymbol{f}_{patho}(\boldsymbol{x}(k))$
                    **for** $targ_i \in c_{targ}$ **do**
                        **for** $v_j \in V$ **do**
                            **if** $v_j = targ_i$ **then**
                                $x_j(k+1) = moda_i$
                            **end if**
                        **end for**
                    **end for**
                    **if** $\exists k' \in [\![1, k]\!] : \boldsymbol{x}(k+1) = \boldsymbol{x}(k')$ **then**
                        $A_{patho} = A_{patho} \cup \{(\boldsymbol{x}(k'), \ldots, \boldsymbol{x}(k))\}$
                        **break**
                    **end if**
                    $k = k + 1$
                **end while**
```

```
                    $H = H \cup \{(\boldsymbol{x}(1), \dots, \boldsymbol{x}(k))\}$
                end for
                if $A_{patho} \subseteq A_{physio}$ then
                    if $A_{patho} = A_{physio}$ then
                        $golden\_set = golden\_set \cup \{(c_{targ}, c_{moda})\}$
                    else
                        $silver\_set = silver\_set \cup \{(c_{targ}, c_{moda})\}$
                    end if
                end if
            end for
        end for
    end for
output $golden\_set$
output $silver\_set$
```

2.5 Working example

To test the algorithm and to illustrate its capacities, it was applied on a published boolean model of the mammalian cell cycle [21]. This model was chosen for several reasons: i) the authors have done a synchronous computation of their boolean network: synchronous boolean networks are the study objects of the present work, ii) their boolean network models a mammalian biological system: the closer to human physiology the model is the better it illustrates the intended applications, iii) the cell cycle is a biological system at the heart of cancer, a widespread and severe disease in human health: this makes the working example more relevant, iv) their network comprises ten nodes: easily computable in the face of the corresponding state space and v) the authors have already computed the attractors of their boolean network: useful to validate the algorithm in computing attractors.

Below are the boolean functions describing their boolean network where, for the sake of readability, x_i stands for $x_i(k)$ and x_{i+} stands for $x_i(k+1)$.

$$
\begin{aligned}
CycD_{+} &= CycD \\
Rb_{+} &= (\neg CycD \wedge \neg CycE \wedge \neg CycA \wedge \neg CycB) \vee (p27 \wedge \neg CycD \wedge \neg CycB) \\
E2F_{+} &= (\neg Rb \wedge \neg CycA \wedge \neg CycB) \vee (p27 \wedge \neg Rb \wedge \neg CycB) \\
CycE_{+} &= E2F \wedge \neg Rb \\
CycA_{+} &= (E2F \wedge \neg Rb \wedge \neg Cdc20 \wedge \neg (Cdh1 \wedge UbcH10)) \vee (CycA \wedge \neg Rb \wedge \neg Cdc20 \wedge \neg (Cdh1 \wedge UbcH10)) \\
p27_{+} &= (\neg CycD \wedge \neg CycE \wedge \neg CycA \wedge \neg CycB) \vee (p27 \wedge \neg (CycE \wedge CycA) \wedge \neg CycB \wedge \neg CycD) \\
Cdc20_{+} &= CycB \\
Cdh1_{+} &= (\neg CycA \wedge \neg CycB) \vee Cdc20 \vee (p27 \wedge \neg CycB) \\
UbcH10_{+} &= \neg Cdh1 \vee (Cdh1 \wedge UbcH10 \wedge (Cdc20 \vee CycA \vee CycB)) \\
CycB_{+} &= \neg Cdc20 \wedge \neg Cdh1
\end{aligned}
$$

Having the example network, two variants of it are needed: the physiological one and the pathological one. The physiological variant is the network as is while the pathological variant is the network plus a constitutive activation or inactivation of at least one of its entities. For simplicity, and given the relatively small number of entities in the network, only one was chosen: the retinoblastoma protein *Rb* for which a constitutive inactivation is applied. To implement this in the pathological variant, the corresponding f_i becomes $Rb_{+} = 0$ in $\boldsymbol{f}_{patho}$. *Rb* was chosen because its inactivation occurs in many cancers [30] and hence a network bearing a constitutive inactivation of it should be a good example of a pathological variant.

2.6 Implementation

The algorithm was firstly implemented in Octave[2] for convenience and then in Fortran 95 compiled with GFortran[3] for performance. The sources can be found on GitHub[4] at `https://github.com/arnaudporet/kali-targ`.

3 Results and discussion

This section exposes the results of the five steps of the algorithm applied to the physiological and pathological variants of the working example.

3.1 Results of step 1

Thanks to the relatively small $size(S)$ of the working example, $size(D)$ was set to $size(S) = 1024$. The algorithm returned the two following attractors:

$$a_1 = \begin{array}{r|ccccccc} CycD: & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ Rb: & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ E2F: & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ CycE: & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ CycA: & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ p27: & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Cdc20: & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ Cdh1: & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ UbcH10: & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ CycB: & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}$$

$$a_2 = \begin{array}{r|c} CycD: & 0 \\ Rb: & 1 \\ E2F: & 0 \\ CycE: & 0 \\ CycA: & 0 \\ p27: & 1 \\ Cdc20: & 0 \\ Cdh1: & 1 \\ UbcH10: & 0 \\ CycB: & 0 \end{array}$$

The attractors are represented as matrices where, for an attractor of length q, lines correspond to the $x_i(k), k \in [\![1, q]\!]$ and where columns correspond to the $\boldsymbol{x}(k)$. Hence $A_{physio} = \{a_1, a_2\}$ which corresponds to what the authors have computed. By the way, a_1 and a_2 are the two physiological attractors. In term of phenotypes, a_1 corresponds to the accomplishment of a normal cell cycle while a_2 corresponds to quiescence.

[2]`https://www.gnu.org/software/octave/`
[3]`https://www.gnu.org/software/gcc/fortran/`
[4]`https://github.com/`

3.2 Results of steps 2 to 5

The results of the steps 2 to 5 were grouped because only the therapeutic bullets found in the step 4 and classified in the step 5 are returned by the algorithm. The algorithm was launched with $r_{min} = 1$ and $r_{max} = 3$. Again due to the relatively small size of the working example, max_{targ} and max_{moda} were set to their maximum over $[\![r_{min}; r_{max}]\!]$, that is $max_{targ} = 120$ and $max_{moda} = 8$. The algorithm returned the following therapeutic bullets:

$$\begin{array}{lll}
+CycD & & silver \\
+CycD & -p27 & silver \\
-CycD & +Rb & silver \\
+CycD & -Rb & silver
\end{array}$$

where $+$ means a therapeutic activation and $-$ a therapeutic inactivation. It should be noted that no golden bullets were found, a not surprising result since they are rarer than silver ones. Furthermore, given these results, the therapeutic activation of *Rb* alone, which was pathologically inactivated, is not enough to remove the pathological attractors. Indeed, as seen in the third bullet, the therapeutic activation of *Rb* must be accompanied by a therapeutic inactivation of *CycD*.

To better illustrate what was done to obtain these therapeutic bullets, here is A_{patho} without the effect of a bullet:

$$a_3 = \begin{array}{r|cccccccc}
CycD: & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
Rb: & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
E2F: & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
CycE: & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
CycA: & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
p27: & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
Cdc20: & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
Cdh1: & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
UbcH10: & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
CycB: & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{array}$$

$$a_4 = \begin{array}{r|ccccccc}
CycD: & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
Rb: & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
E2F: & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
CycE: & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
CycA: & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
p27: & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
Cdc20: & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
Cdh1: & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
UbcH10: & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
CycB: & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{array}$$

$a_4 = a_1 \in A_{physio}$: a_4 is a physiological attractor. Indeed, it is possible that the pathological variant produces physiological attractors: A_{patho} is not the set containing exactly all the pathological attractors, it is the attractor set of the pathological variant and it is possible that $A_{physio} \cap A_{patho} \neq \emptyset$. However $a_3 \notin A_{physio}$: it is a pathological attractor. In term of phenotypes, a_3 may correspond to a somewhat degenerated cell cycle and is what a therapeutic bullet, being golden or silver, is intended to avoid.

Again to better illustrate what was done to obtain the therapeutic bullets, here is A_{patho} under the effect of the third bullet:

$$\begin{array}{rl} CycD: & 0 \\ Rb: & 1 \\ E2F: & 0 \\ CycE: & 0 \\ CycA: & 0 \\ p27: & 1 \\ Cdc20: & 0 \\ Cdh1: & 1 \\ UbcH10: & 0 \\ CycB: & 0 \end{array}$$

which is a_2. Hence, as expected for a therapeutic bullet, the pathological attractor a_3 was removed, but the physiological attractor a_1 was not restored: the third bullet is then a silver one. Consequently, with this bullet, no cell cycle occurs, being normal or degenerated, and the only reachable phenotype is quiescence. While disabling the cell cycle of cancer cells is beneficial to the ill organism, what about disabling the cell cycle of its normal cell. As mentioned above, with silver bullets this is a matter of choice between a silver bullet and no therapeutic bullet.

4 Conclusions

Under the assumption stating that dynamical system attractors and biological network phenotypes are linked when the first one models the second one, the results showed that the algorithm succeed in performing the proposed *in silico* target identification by returning four effective therapeutic bullets for a pathological variant of the mammalian cell cycle relevant in diseases such as cancer. Consequently, it can be used on other synchronous boolean models of biological networks involved in disease pathophysiologies for *in silico* target identification provided that the physiological and pathological variants are known. However, one should keep in mind that the findings produced by the algorithm are conditional upon the validity of the assumption linking attractors and phenotypes.

It should be noted that target identification, being *in silico* or not, is a piece belonging to a wider machinery: target discovery. Indeed, having an *in silico* demonstrated potential target implies that further work has to be done to obtain a drug which will be effective *in vivo*. Among other characteristics, such a drug has to be absorbed by the organism, has to reach its target and has to be non-toxic at therapeutic dosages. Furthermore, as with any *in silico* evidence, it should be validated in practice: there is a bridge to cross between theory and practice. Nevertheless, the present algorithm is expected to find a place in, and to contribute to, the endeavors done in therapeutic researches.

Regarding further investigations that could be done on this one, extending the algorithm for asynchronous boolean networks is important because such models are likely to better approach the dynamic of biological systems [31]. Indeed, in a biological system, events can be subjected to stochasticity, they do not necessarily occur simultaneously or they do not belong to the same time scale, three things that a synchronous updating does not take into account. Furthermore, one of the limitations of boolean models is that their variables can take only two values. However, in reality, things are not necessarily binary and variables should be able to take much more values for more accuracy. Without leaving the logic-based framework, one solution is to use more general forms of logic such as multivalued logic or fuzzy logic [32].

References

[1] Mark A Lindsay. Target discovery. *Nature Reviews Drug Discovery*, 2(10):831–838, 2003.

[2] Jonathan Knowles and Gianni Gromo. Target selection in drug discovery. *Nature Reviews Drug Discovery*, 2(1):63–69, 2003.

[3] Peter Imming, Christian Sinning, and Achim Meyer. Drugs, their targets and the nature and number of drug targets. *Nature reviews Drug discovery*, 5(10):821–834, 2006.

[4] Grant R Zimmermann, Joseph Lehar, and Curtis T Keith. Multi-target therapeutics: when the whole is greater than the sum of the parts. *Drug discovery today*, 12(1):34–42, 2007.

[5] Eliot H Ohlstein, Robert R Ruffolo Jr, and John D Elliott. Drug discovery in the next millennium. *Annual review of pharmacology and toxicology*, 40(1):177–191, 2000.

[6] Jackson B Gibbs. Mechanism-based target identification and drug discovery in cancer research. *Science*, 287(5460):1969–1973, 2000.

[7] KI Kaitin. Deconstructing the drug development process: the new face of innovation. *Clinical Pharmacology & Therapeutics*, 87(3):356–361, 2010.

[8] Xiao Hua Ma, Zhe Shi, Chunyan Tan, Yuyang Jiang, Mei Lin Go, Boon Chuan Low, and Yu Zong Chen. In-silico approaches to multi-target drug discovery. *Pharmaceutical research*, 27(5):739–749, 2010.

[9] Denis Noble, Jeremy Levin, and William Scott. Biological simulations in drug discovery. *Drug Discovery Today*, 4(1):10–16, 1999.

[10] Xin Chen, Zhi Liang Ji, and Yu Zong Chen. Ttd: therapeutic target database. *Nucleic acids research*, 30(1):412–415, 2002.

[11] M Whirl-Carrillo, EM McDonagh, JM Hebert, L Gong, K Sangkuhl, CF Thorn, RB Altman, and Teri E Klein. Pharmacogenomics knowledge for personalized medicine. *Clinical Pharmacology & Therapeutics*, 92(4):414–417, 2012.

[12] Minoru Kanehisa and Susumu Goto. Kegg: kyoto encyclopedia of genes and genomes. *Nucleic acids research*, 28(1):27–30, 2000.

[13] David Croft, Gavin O'Kelly, Guanming Wu, Robin Haw, Marc Gillespie, Lisa Matthews, Michael Caudy, Phani Garapati, Gopal Gopinath, Bijay Jassal, et al. Reactome: a database of reactions, pathways and biological processes. *Nucleic acids research*, 39(suppl 1):D691–D697, 2011.

[14] David S Wishart, Craig Knox, An Chi Guo, Dean Cheng, Savita Shrivastava, Dan Tzur, Bijaya Gautam, and Murtaza Hassanali. Drugbank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic acids research*, 36(suppl 1):D901–D906, 2008.

[15] Hiroaki Kitano. Systems biology: a brief overview. *Science*, 295(5560):1662–1664, 2002.

[16] Hiroaki Kitano. Computational systems biology. *Nature*, 420(6912):206–210, 2002.

[17] Barbara Di Ventura, Caroline Lemerle, Konstantinos Michalodimitrakis, and Luis Serrano. From in vivo to in silico biology and back. *Nature*, 443(7111):527–533, 2006.

[18] Stefan Bornholdt. Boolean network models of cellular regulation: prospects and limitations. *Journal of the Royal Society Interface*, 5(Suppl 1):S85–S94, 2008.

[19] Sui Huang and Donald E Ingber. Shape-dependent control of cell growth, differentiation, and apoptosis: switching between attractors in cell regulatory networks. *Experimental cell research*, 261(1):91–103, 2000.

[20] Maria I Davidich and Stefan Bornholdt. Boolean network model predicts cell cycle sequence of fission yeast. *PLoS One*, 3(2):e1672, 2008.

[21] Adrien Fauré, Aurélien Naldi, Claudine Chaouiya, and Denis Thieffry. Dynamical analysis of a generic boolean model for the control of the mammalian cell cycle. *Bioinformatics*, 22(14):e124–e131, 2006.

[22] Herman F Fumiã and Marcelo L Martins. Boolean network model for cancer pathways: Predicting carcinogenesis and targeted therapy outcomes. *PloS one*, 8(7):e69008, 2013.

[23] Pau Creixell, Erwin M Schoof, Janine T Erler, and Rune Linding. Navigating cancer network attractors for tumor-specific therapy. *Nature biotechnology*, 30(9):842–848, 2012.

[24] Keith Baverstock. A comparison of two cell regulatory models entailing high dimensional attractors representing phenotype. *Progress in Biophysics and Molecular Biology*, 106(2):443–449, 2011.

[25] Michelle L Wynn, Nikita Consul, Sofia D Merajver, and Santiago Schnell. Logic-based models in systems biology: a predictive and parameter-free network analysis method. *Integrative Biology*, 4(11):1323–1337, 2012.

[26] Abhishek Garg, Alessandro Di Cara, Ioannis Xenarios, Luis Mendoza, and Giovanni De Micheli. Synchronous versus asynchronous modeling of gene regulatory networks. *Bioinformatics*, 24(17):1917–1925, 2008.

[27] Xiaowei Zhu, Mark Gerstein, and Michael Snyder. Getting connected: analysis and principles of biological networks. *Genes & development*, 21(9):1010–1024, 2007.

[28] Assieh Saadatpour and Réka Albert. Boolean modeling of biological regulatory networks: a methodology tutorial. *Methods*, 2012.

[29] Desheng Zheng, Guowu Yang, Xiaoyu Li, Zhicai Wang, Feng Liu, and Lei He. An efficient algorithm for computing attractors of synchronous and asynchronous boolean networks. *PloS one*, 8(4):e60593, 2013.

[30] Charles J Sherr and Frank McCormick. The rb and p53 pathways in cancer. *Cancer cell*, 2(2):103–112, 2002.

[31] Assieh Saadatpour, István Albert, and Réka Albert. Attractor analysis of asynchronous boolean models of signal transduction networks. *Journal of theoretical biology*, 266(4):641–656, 2010.

[32] Lotfi Asker Zadeh. Fuzzy logic. *Computer*, 21(4):83–93, 1988.