A Decomposition-based Approach towards the Control of Boolean Networks

Soumya Paul
SnT, University of Luxembourg

Cui Su
SnT, University of Luxembourg

Jun Pang
SnT & FSTC, University of Luxembourg

Andrzej Mizera
LCSB, University of Luxembourg
DII, Luxembourg Institute of Health

ABSTRACT

We study the problem of computing a minimal subset of nodes of a given asynchronous Boolean network that need to be controlled to drive its dynamics from an initial steady state (or *attractor*) to a target steady state. Due to the phenomenon of state-space explosion, a simple global approach that performs computations on the entire network, may not scale well for large networks. We believe that efficient algorithms for such networks must exploit the structure of the networks together with their dynamics. Taking such an approach, we derive a decomposition-based solution to the minimal control problem which can be significantly faster than the existing approaches on large networks. We apply our solution to both real-life biological networks and randomly generated networks, demonstrating promising results.

KEYWORDS

Boolean networks, attractors, network control, decomposition

ACM Reference Format:
Soumya Paul, Cui Su, Jun Pang, and Andrzej Mizera. 2018. A Decomposition-based Approach towards the Control of Boolean Networks. In *ACM-BCB'18: 9th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics, August 29-September 1, 2018, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3233547.3233550

1 INTRODUCTION

Cell reprogramming is a way to change one cell phenotype to another, allowing tissue or neuron regeneration techniques. Recent studies have shown that differentiated adult cells can be reprogrammed to embryonic-like pluripotent state or directly to other types of adult cells without the need of intermediate reversion to pluripotent state [5, 20]. This has led to a surge in regenerative medicine and there is a growing need for the discovery of new and efficient methods for the control of cellular behaviour.

In this work we focus on the study and control of gene regulatory networks (GRNs) and their combined dynamics with an associated signalling pathway. GRNs are graphical diagrams visualising the relationships between genes and their regulators. They represent biological systems characterised by the orchestrated interplay of complex interactions resulting in highly nested feedback and feed-forward loops. Signalling networks consist of interacting signalling pathways that perceive the changes in the environment and allow the cell to correctly respond to them by appropriately adjusting its gene-expression. These pathways are often complex, multi-component biological systems that are regulated by various feedbacks and that interfere with each other via diverse cross-talks. As a result, GRNs with integrated signalling networks are representatives of complex systems characterised by non-linear dynamics. These factors render the design of external control strategies for these biological systems a very challenging task. So far, no general mathematical frameworks for the control of this type of systems have been developed [3, 8, 9].

Boolean networks (BNs), first introduced by Kauffman [7], is a popular and well-established framework for modelling GRNs and their associated signalling pathways. Its main advantage is that it is simple and yet able to capture the important dynamic properties of the system under study, thus facilitating the modelling of large biological systems as a whole. The states of a BN are tuples of 0s and 1s where each element of the tuple represents the level of activity of a particular protein in the GRN or the signalling pathway it models - 0 for inactive and 1 for active. The BN is assumed to evolve dynamically by moving from one state to the next governed by a Boolean function for each of its components. The steady state behaviour of a BN is given by its subset of states called *attractors* to one of which the dynamics eventually settles down. In biological context, attractors are hypothesised to characterise cellular phenotypes [7] and also correspond to functional cellular states such as proliferation, apoptosis differentiation etc. [6].

Cellular reprogramming, or the control of the GRNs and their signalling pathways therefore amounts to being able to drive the dynamics of the associated BN from an attractor to another 'desirable' target attractor by controlling or reprogramming the nodes of the BN. This needs to be done while respecting certain constraints viz. a minimal subset of nodes of the BN are controlled or the control is applied only for a minimal number of time steps. Under such constraints, it is known that the problem of driving the BN from a source to a target attractor (the control problem) is computationally difficult [11, 12] and does not scale well to large networks. Thus a simple global approach (see Section 3.4 for a description) treating the entire network in one-go is highly inefficient. This is intuitively due to the infamous state-space explosion problem. Since most

ACM-BCB'18, August 29-September 1, 2018, Washington, DC, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5794-4/18/08…$15.00
https://doi.org/10.1145/3233547.3233550

real-life networks are large, there is a strong need for designing algorithms which exploit certain properties (structural or dynamic or both) of a BN and can efficiently address the control problem.

Our contributions. In this paper, we develop a generic approach towards solving the minimal control problem (defined formally in Section 3) on large BNs based on combining both their structural and the dynamic properties. We show that:

- The problem of computing the minimal set of nodes to be controlled in a single time-step (simultaneously) to drive the system from a source state $\mathbf{s}$ to a target attractor A_t (*driver nodes*) is equivalent to computing a subset of states of the state transition graph of the BN called the *strong basin* (defined in Section 3) of attraction of A_t (dynamic property).
- We show how the network structure of a large BN can be explored to decompose it into smaller *blocks*. The strong basins of attractions of the projection of A_t to these blocks can be computed *locally* and then combined to recover the *global* strong basin of attraction of A_t (structural property).
- Any algorithm for the computation of the global strong basin of attraction of A_t can also be used to compute the local strong basins of attraction of the projections of A_t to the blocks of BN. Doing so results in the improvement in efficiency for certain networks which have modular structures (like most real-life biological networks).
- We concretise our approach by describing in detail one such algorithm (Algorithm 1) which is based on the computation of fixed points of set operations.
- We have implemented our decomposition-based approach using this algorithm and applied it to a number of case studies of BNs corresponding to real-life biological networks and randomly generated BNs. Our results show that for certain structurally well-behaved BNs our decomposition-based approach is efficient and outperforms the global approach.

2 RELATED WORK

In recent years, several approaches have been developed for the control of complex networks [1–3, 9, 11, 12, 15, 22–24]. Among them, the methods [1, 3, 9] were proposed to tackle the control of networks with linear time-invariant dynamics. Liu et al. [9] first developed a structural controllability framework for complex networks to solve full control problems, by identifying the minimal set of (driver) nodes that can steer the entire dynamics of the system. Afterwards, Gao et al. extended this method to the target control of complex networks [3]. They proposed a k-walk method and a greedy algorithm to identify a set of driver nodes for controlling a pre-selected set of target nodes. However, Czeizler et al. [1] proved that it is NP-hard to find the minimal set of driver nodes for structural target control problems and they improved the greedy algorithm [3] using several heuristics. The above methods have a common distinctive advantage that they are solely based on the network structures, which are exponentially smaller than the number of states in their dynamics. Nevertheless, they are only applicable to systems with linear time-invariant dynamics.

The control methods proposed in [2, 11, 12, 15, 22–24] are designed for networks governed by non-linear dynamics. Among these methods, the ones based on the computation of the feedback vertex set (FVS) [2, 15, 24] and the 'stable motifs' of the network [23] drive the network towards a target state by regulating a component of the network with some constraints (feedback vertex sets and stable motifs). The method based on FVS is purely a structure-based method, while that based on stable motifs takes into account the functional information of the network (network dynamics) and has a substantial improvement in computing the number of driver nodes. These two methods are promising, even though none of them guarantees to find the minimal set of driver nodes. In [22], Wang et al. highlighted an experimentally feasible approach towards the control of nonlinear dynamical networks by constructing 'attractor networks' that reflect their controllability. They construct the attractor network of a system by including all the experimentally validated paths between the attractors of the network. The concept of an attractor network is inspiring. However, this method cannot provide a straightforward way to find the paths from one attractor to a desired attractor, and it fails to formulate a generic framework for the control of nonlinear dynamical networks.

Closely related to our work, Mandon et al. [11, 12] proposed approaches towards the control of asynchronous BNs. In particular, in [11] they proposed a few algorithms to identify reprogramming determinants for both existential and inevitable reachability of the target attractor with permanent perturbations. Later on, they proposed an algorithm that can find all existing control paths between two states within a limited number of either permanent or temporary perturbations [12]. However, these methods do not scale well for large networks.[1] This is mainly due to the fact that they need to encode all possible control strategies into the transition system of the BN in order to identify the desired reprogramming paths [12]. As a consequence, the size of the resulting *perturbed transition graph* grows exponentially with the number of allowed perturbations, which renders their algorithms inefficient.

The identified limitations of these existing approaches motivate us to develop a new approach towards the control of non-linear Boolean networks which is modular and exploits *both* their structural and dynamic properties. Gates et al. [4] showed that such an approach is inevitable for the identification of the correct parameters and control strategies, in that, focussing only on a single property (either structural or dynamic) might lead to both their overestimation or underestimation.

3 PRELIMINARIES

3.1 Boolean networks

A Boolean network (BN) describes elements of a dynamical system with binary-valued nodes and interactions between elements with Boolean functions. It is formally defined as:

Definition 3.1 (Boolean networks). A Boolean network is a tuple $\mathsf{BN} = (\mathbf{x}, \mathbf{f})$ where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ such that each $x_i, 1 \leq i \leq n$ is a Boolean variable and $\mathbf{f} = (f_1, f_2, \ldots, f_n)$ is a tuple of Boolean functions over $\mathbf{x}$. $|\mathbf{x}| = n$ denotes the number of variables.

In what follows, i will always range between 1 and n, unless stated otherwise. A Boolean network $\mathsf{BN} = (\mathbf{x}, \mathbf{f})$ may be viewed as a directed graph $\mathcal{G}_{\mathsf{BN}} = (V, E)$ where $V = \{v_1, v_2 \ldots, v_n\}$ is the set

[1] We learnt through private communication that the current implementation of their methods does not scale efficiently to BNs having more than 20 nodes

of vertices or nodes and for every $1 \leq i, j \leq n$, there is a directed edge from v_j to v_i if and only if f_i depends on x_j. An edge from v_j to v_i will be often denoted as $v_j \rightarrow v_i$. A path from a vertex v to a vertex v' is a (possibly empty) sequence of edges from v to v' in $\mathcal{G}_{\mathsf{BN}}$. For any vertex $v \in V$ we define its set of parents as $\mathrm{par}(v) = \{v' \in V \mid v' \rightarrow v\}$. For the rest of the exposition, we assume that an arbitrary but fixed network BN of n variables is given to us and $\mathcal{G}_{\mathsf{BN}} = (V, E)$ is its associated directed graph.

A state $\mathbf{s}$ of BN is an element in $\{0,1\}^n$. Let S be the set of states of BN. For any state $\mathbf{s} = (s_1, s_2, \ldots, s_n)$, and for every i, the value of s_i, often denoted as $\mathbf{s}[i]$, represents the value that the variable x_i takes when the BN 'is in state $\mathbf{s}$'. For some i, suppose f_i depends on $x_{i_1}, x_{i_2}, \ldots, x_{i_k}$. Then $f_i(\mathbf{s})$ will denote the value $f_i(\mathbf{s}[i_1], \mathbf{s}[i_2], \ldots, \mathbf{s}[i_k])$. For two states $\mathbf{s}, \mathbf{s}' \in \mathsf{S}$, the Hamming distance between $\mathbf{s}$ and $\mathbf{s}'$ will be denoted as $\mathrm{hd}(\mathbf{s}, \mathbf{s}')$ and $\arg(\mathrm{hd}(\mathbf{s}, \mathbf{s}')) \subseteq \{1, 2, \ldots, n\}$ will denote the set of indices in which $\mathbf{s}$ and $\mathbf{s}'$ differ. For a state $\mathbf{s}$ and a subset $\mathsf{S}' \subseteq \mathsf{S}$, the Hamming distance between $\mathbf{s}$ and S' is defined as the minimum of the Hamming distances between $\mathbf{s}$ and all the states in S'. That is, $\mathrm{hd}(\mathbf{s}, \mathsf{S}') = \min_{\mathbf{s}' \in \mathsf{S}'} \mathrm{hd}(\mathbf{s}, \mathbf{s}')$. We let $\arg(\mathrm{hd}(\mathbf{s}, \mathsf{S}'))$ denote the set of subsets of $\{1, 2, \ldots, n\}$ such that $I \in \arg(\mathrm{hd}(\mathbf{s}, \mathsf{S}'))$ if and only if I is a set of indices of the variables that realise this Hamming distance.

3.2 Dynamics of Boolean networks

We assume that the Boolean network evolves in discrete time steps. It starts initially in a state $\mathbf{s}_0$ and its state changes in every time step according to the update functions $\mathbf{f}$. The updating may happen in various ways. Every such way of updating gives rise to a different dynamics for the network. In this work, we shall be interested primarily in the asynchronous updating scheme.

Definition 3.2 (Asynchronous dynamics of Boolean networks). Suppose $\mathbf{s}_0 \in \mathsf{S}$ is an initial state of BN. The asynchronous evolution of BN is a function $\xi : \mathbb{N} \rightarrow \wp(\mathsf{S})$ such that $\xi(0) = \mathbf{s}_0$ and for every $j \geq 0$, if $\mathbf{s} \in \xi(j)$ then $\mathbf{s}' \in \xi(j+1)$ if and only if either $\mathrm{hd}(\mathbf{s}, \mathbf{s}') = 1$ and $\mathbf{s}'[i] = f_i(\mathbf{s})$ where $i = \arg(\mathrm{hd}(\mathbf{s}, \mathbf{s}'))$ or $\mathrm{hd}(\mathbf{s}, \mathbf{s}') = 0$ and there exists i such that $\mathbf{s}'[i] = f_i(\mathbf{s})$.

Note that the asynchronous dynamics is non-deterministic – the value of exactly one variable is updated in a single time-step. The index of the variable that is updated is not known in advance. Henceforth, when we talk about the dynamics of BN, we shall mean the asynchronous dynamics as defined above.

The dynamics of a Boolean network can be represented as a *state transition graph* or a *transition system (TS).*

Definition 3.3 (Transition system of BN). The transition system of BN, denoted by the generic notation TS is a tuple $(\mathsf{S}, \rightarrow)$ where the vertices are the set of states S and for any two states $\mathbf{s}$ and $\mathbf{s}'$ there is a directed edge from $\mathbf{s}$ to $\mathbf{s}'$, denoted $\mathbf{s} \rightarrow \mathbf{s}'$ if and only if either $\mathrm{hd}(\mathbf{s}, \mathbf{s}') = 1$ and $\mathbf{s}'[i] = f_i(\mathbf{s})$ where $i = \arg(\mathrm{hd}(\mathbf{s}, \mathbf{s}'))$ or $\mathrm{hd}(\mathbf{s}, \mathbf{s}') = 0$ and there exists i such that $\mathbf{s}'[i] = f_i(\mathbf{s})$.

3.3 Attractors and basins of attraction

A path from a state $\mathbf{s}$ to a state $\mathbf{s}'$ is a (possibly empty) sequence of transitions from $\mathbf{s}$ to $\mathbf{s}'$ in TS. A path from a state $\mathbf{s}$ to a subset S' of S is a path from $\mathbf{s}$ to any state $\mathbf{s}' \in \mathsf{S}'$. For any state $\mathbf{s} \in \mathsf{S}$, let $\mathrm{pre}_{\mathsf{TS}}(\mathbf{s}) = \{\mathbf{s}' \in \mathsf{S} \mid \mathbf{s}' \rightarrow \mathbf{s}\}$ and let $\mathrm{post}_{\mathsf{TS}}(\mathbf{s}) = \{\mathbf{s}' \in \mathsf{S} \mid \mathbf{s} \rightarrow \mathbf{s}'\}$. $\mathrm{pre}_{\mathsf{TS}}(\mathbf{s})$ contains all the states that can reach $\mathbf{s}$ by performing a single transition in TS and $\mathrm{post}_{\mathsf{TS}}(s)$ contains all the states that can be reached from $\mathbf{s}$ by a single transition in TS. Note that, by definition, $\mathrm{hd}(\mathbf{s}, \mathrm{pre}_{\mathsf{TS}}(\mathbf{s})) \leq 1$ and $\mathrm{hd}(\mathbf{s}, \mathrm{post}_{\mathsf{TS}}(\mathbf{s})) \leq 1$. $\mathrm{pre}_{\mathsf{TS}}$ and $\mathrm{post}_{\mathsf{TS}}$ can be lifted to a subset S' of S as: $\mathrm{pre}_{\mathsf{TS}}(\mathsf{S}') = \bigcup_{\mathbf{s} \in \mathsf{S}'} \mathrm{pre}_{\mathsf{TS}}(\mathbf{s})$ and $\mathrm{post}_{\mathsf{TS}}(\mathsf{S}') = \bigcup_{\mathbf{s} \in \mathsf{S}'} \mathrm{post}_{\mathsf{TS}}(\mathbf{s})$. For a state $\mathbf{s} \in \mathsf{S}$, $\mathrm{reach}_{\mathsf{TS}}(\mathbf{s})$ denotes the set of states $\mathbf{s}'$ such that there is a path from $\mathbf{s}$ to $\mathbf{s}'$ in TS and can be defined as the transitive closure of the $\mathrm{post}_{\mathsf{TS}}$ operation. Thus, $\mathrm{reach}_{\mathsf{TS}}(\mathbf{s})$ is the smallest subset of states in S such that $\mathbf{s} \in \mathrm{reach}_{\mathsf{TS}}(\mathbf{s})$ and $\mathrm{post}_{\mathsf{TS}}(\mathrm{reach}_{\mathsf{TS}}(\mathbf{s})) \subseteq \mathrm{reach}_{\mathsf{TS}}(\mathbf{s})$.

Definition 3.4 (Attractor). An attractor A of TS (or of BN) is a subset of states of S such that for every $\mathbf{s} \in A$, $\mathrm{reach}_{\mathsf{TS}}(\mathbf{s}) = A$.

Any state which is not part of an attractor is a transient state. An attractor A of TS is said to be reachable from a state $\mathbf{s}$ if $\mathrm{reach}_{\mathsf{TS}}(\mathbf{s}) \cap A \neq \emptyset$. Attractors represent the stable behaviour of the BN according to the dynamics. The network starting at any initial state $\mathbf{s}_0 \in \mathsf{S}$ will eventually end up in one of the attractors of TS and remain there forever unless perturbed.

Observation 1. Any attractor of TS is a bottom strongly connected component of TS.

For an attractor A of TS, we define subsets of states of S called the weak and strong basins of attractions of A, denoted as $\mathrm{bas}^{W}_{\mathsf{TS}}(A)$ and $\mathrm{bas}^{S}_{\mathsf{TS}}(A)$, respectively, as follows.

Definition 3.5 (Basin of attraction). Let A be an attractor of TS.

- **Weak basin:** The weak basin of attraction of A with respect to TS, is defined as $\mathrm{bas}^{W}_{\mathsf{TS}}(A) = \{\mathbf{s} \in \mathsf{S} \mid \mathrm{reach}_{\mathsf{TS}}(\mathbf{s}) \cap A \neq \emptyset\}$.
- **Strong basin:** The strong basin of attraction of A with respect to TS, is defined as $\mathrm{bas}^{S}_{\mathsf{TS}}(A) = \mathrm{bas}^{W}_{\mathsf{TS}}(A) \setminus \mathrm{bas}^{W}_{\mathsf{TS}}(A')$ where A' is an attractor of TS and $A' \neq A$.

Thus the weak basin of attraction of A is the set of all states $\mathbf{s}$ from which there is a path to A. It is possible that there are paths from $\mathbf{s}$ to some other attractor $A' \neq A$. However, the notion of a strong basin does not allow this. Thus, if $\mathbf{s} \in \mathrm{bas}^{S}_{\mathsf{TS}}(A)$ then $\mathbf{s} \notin \mathrm{bas}^{W}_{\mathsf{TS}}(A')$ for any other attractor A'. We need the notion of strong basin to *ensure* reachability to the target attractor after applying control.

Example 3.6. Consider the three-node network $\mathsf{BN} = (\mathbf{x}, \mathbf{f})$ where $\mathbf{x} = (x_1, x_2, x_3)$ and $\mathbf{f} = (f_1, f_2, f_3)$ where $f_1 = \neg x_2 \vee (x_1 \wedge x_2)$, $f_2 = x_1 \wedge x_2$ and $f_3 = (\neg x_1 \wedge x_2) \vee (\neg x_2 \wedge x_3)$. The graph of the network $\mathcal{G}_{\mathsf{BN}}$ and its associated transition system TS is given in Figure 1. TS has three attractors $\{(100)\}$, $\{(110)\}$ and $\{(101)\}$ shown by dark grey rectangles. Their corresponding strong basins of attractions are shown by enclosing grey regions of a lighter shade. Note that, there is a path from the state (010) to both the attractors $\{(100)\}$ and $\{(101)\}$. Hence (010) is not in the strong basin of either of these attractors but is in the weak basins of both of them.

Observation 2. Given an attractor A, we can compute the weak basin $\mathrm{bas}^{W}_{\mathsf{TS}}(A)$ by an iterative fixpoint procedure. Indeed, $\mathrm{bas}^{W}_{\mathsf{TS}}(A)$ is the smallest subset W of S such that $A \in W$ and $\mathrm{pre}_{\mathsf{TS}}(W) \subseteq W$. We shall call this procedure Compute_Weak_Basin which will take as arguments the function tuple $\mathbf{f}$ and an attractor A.

Henceforth, to avoid clutter, we shall drop the subscript TS when the transition system is clear from the context. Also, we shall often

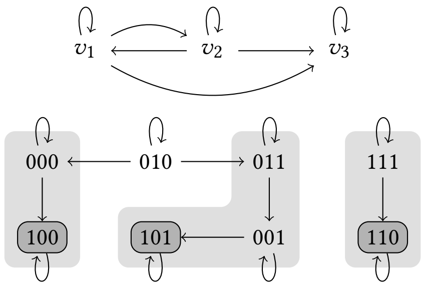


Figure 1: The graph of BN **and its transition system.**

drop the superscript S as well the mention of the word "strong" when dealing with strong basins. Thus the "basin of A" will always mean the strong basin of attraction of A unless mentioned otherwise and will be denoted as $\mathrm{bas}(A)$.

3.4 The control problem

As described in the introduction, the attractors of a Boolean network represent the cellular phenotypes, the expressions of the genes etc. Some of these attractors may be diseased, weak or undesirable while others are healthy and desirable. Curing a disease is thus in effect, moving the dynamics of the network from an undesired 'source' attractor to a desired 'target' attractor.

One of the ways to achieve the above is by controlling the various 'parameters' of the network, for eg. the values of the variables, or the Boolean functions themselves. In this exposition, we shall be interested in the former kind of control, that is, tweaking the values of the variables of the network. Such a control may be (i) *permanent* – the value(s) of one or more variables are fixed forever, for all the following time steps or (ii) *temporary* – the values of (some of) the variables are fixed for a finite number (one or more) of time steps and then the control is removed to let the system evolve on its own. Moreover, the variables can be either controlled (a) *simultaneously* – the control is applied to all the variables at once or (b) *sequentially* – the control is applied over a sequence of steps.

In this work we shall be interested in the control of type (ii) and (a). Moreover, for us, the perturbations are applied only for a *single* time step. Thus we can formally define control as follows.

Definition 3.7 (Control). A control C is a (possibly empty) subset of $\{1, 2, \ldots, n\}$. For a state $\mathbf{s} \in \mathrm{S}$, the application of a control C to $\mathbf{s}$, denoted $\mathrm{C}(\mathbf{s})$ is defined as the state $\mathbf{s}' \in \mathrm{S}$ such that $\mathbf{s}'[i] = (1 - \mathbf{s}[i])$ if $i \in \mathrm{C}$ and $\mathbf{s}'[i] = \mathbf{s}[i]$ otherwise. Given a control C, the set of vertices $\{v_i \mid i \in \mathrm{C}\}$ of $\mathcal{G}_{\mathrm{BN}}$ will be called the driver nodes for C.

Our aim is to make the control as less invasive to the system as possible. Thus not only is the control applied for just a single time step, it is also applied to as few of the nodes of the Boolean network as possible. The minimal simultaneous single-step target-control problem for Boolean networks that we are thus interested in can be formally stated as follows.

Minimal simultaneous target-control: Given a Boolean network BN, a 'source state' $\mathbf{s} \in \mathrm{S}$ and a 'target attractor' A_t of TS, compute a control C such that after the application of $\mathrm{C}(\mathbf{s})$, BN eventually reaches A_t and C is a minimal such subset of $\{1, 2, \ldots, n\}$. We shall call such a control a *minimal control* from $\mathbf{s}$ to A_t. The set of all minimal controls from $\mathbf{s}$ to A_t will be denoted as $\mathbb{C}_{\mathrm{min}}^{\mathbf{s} \to A_t}$.

Note that the requirement of minimality is crucial, without which the problem is rendered trivial - simply pick some state $\mathbf{s}' \in A_t$ and move to it. The nodes required to be controlled will often be called the driver nodes for the corresponding control. Our goal is to provide an efficient algorithm for the above question. That is, to devise an algorithm that takes as input only the Boolean functions $\mathbf{f}$ of BN, a source state $\mathbf{s}$ and a target attractor A_t of TS and outputs the indices of a minimal subset of nodes of $\mathbf{s}$ that need to be toggled or controlled (the driver nodes) so that after applying the control, the dynamics eventually and surely reaches A_t. It is known that in general the problem is computationally difficult – PSPACE-hard [11] and unless certain open conjectures in computational complexity are false, these questions are computationally difficult and would require time exponential in the size of the Boolean network. That is intuitively because of the infamous state-space explosion phenomenon – the number of states of the transition system is exponential in the network-size.

Observation 3. It is important to note that if the BN is in some state $\mathbf{s} \in \mathrm{bas}(A)$ in some time step t, that is if $\xi(t) = \mathbf{s}$ then by the definition of $\mathrm{bas}(A)$ it will eventually and surely reach a state $\mathbf{s}' \in A$. That is, there exists a time step $t' > t$ such that $\xi(t') = \mathbf{s}'$. Hence given a source state $\mathbf{s}$ and a target attractor A_t, $\mathbb{C}_{\mathrm{min}}^{\mathbf{s} \to A_t}$ can easily be seen to be equal to $\arg(\mathrm{hd}(\mathbf{s}, \mathrm{bas}(A_t)))$. In other words

Proposition 3.8. *A control* C *from* $\mathbf{s}$ *to* A_t *is minimal if and only if* $\mathrm{C}(\mathbf{s}) \in \mathrm{bas}(A_t)$ *and* $\mathrm{C} \in \arg(\mathrm{hd}(\mathbf{s}, \mathrm{bas}(A)))$.

Proof. If $\mathrm{C}(\mathbf{s}) \notin \mathrm{bas}(A_t)$ then either (a) $\mathrm{C}(\mathbf{s}) \notin \mathrm{bas}_{\mathrm{TS}}^{W}(A_t)$ or (b) $\mathrm{C}(\mathbf{s}) \in \mathrm{bas}_{\mathrm{TS}}^{W}(A_t)$. If (a) holds, then there is no path from $\mathrm{C}(\mathbf{s})$ to A_t and if (b) holds, then there is a path from $\mathrm{C}(\mathbf{s})$ to some other attractor $A \neq A_t$. In either case BN is not guaranteed to reach a state in A_t after the control C is applied to $\mathbf{s}$. And, if $\mathrm{C} \notin \arg(\mathrm{hd}(\mathbf{s}, \mathrm{bas}(A)))$, then C cannot be minimal (by definition of Hamming distance), and conversely. □

Thus, solving the minimal simultaneous target-control problem efficiently boils down to how efficiently we can compute the strong basin of the target attractor.

Example 3.9. Continuing with Example 3.6, suppose we are in source state $\mathbf{s} = (101)$ (which is also an attractor) and we want to apply (minimal simultaneous) control to $\mathbf{s}$ so the system eventually and surely moves to the target attractor $A_t = \{(110)\}$. We could flip $\mathbf{s}[2]$ and $\mathbf{s}[3]$ to move directly to A_t which would require a control $\mathrm{C} = \{2, 3\}$. However, if we notice that the state (111) is in the basin of A_t we can simply apply a control $\mathrm{C}' = \{2\}$ and the dynamics of the BN will ensure that it eventually reaches A_t. Indeed, C' is also the minimal control in this case.

3.5 A global algorithm

In the rest of this section, we first describe a procedure for computing the (strong) basin of an attractor based on the computation of fixed point. We then use this procedure to design a simple global algorithm for solving the minimal simultaneous target-control problem based on a global computation of the basin of the target attractor A_t. This algorithm will act as a reference for comparing

Algorithm 1 Fixpoint computation of strong basin

```
1: procedure COMPUTE_SRONG_BASIN($\mathbf{f}$,$A$)
2:     Let WB = COMPUTE_WEAK_BASIN($\mathbf{f}$, $A$)
3:     Initialise SB = $\emptyset$
4:     Till SB $\neq$ WB do
5:         If SB$\neq$ $\emptyset$ do WB = SB
6:         SB=WB\(pre(post(WB) \ WB)$\cap$WB)
7:     done
8:     return SB
9: end procedure
```

Algorithm 2 Global minimal simultaneous target control

```
procedure GLOBAL_MINIMAL_CONTROL($\mathbf{f}$, $\mathbf{s}$, $A_t$)
    Let SB = COMPUTE_STRONG_BASIN($\mathbf{f}$, $A_t$)
    return arg(hd($\mathbf{s}$,SB))
end procedure
```

the decomposition-based algorithm which we shall later develop. We first introduce an algorithm called COMPUTE_STRONG_BASIN, described in Algorithm 1, for the computation of the strong basin of an attractor A based on a fixpoint approach. The most important step of this algorithm is Line 6, which is repeated till the set SB settles down to a fixed point, which is the strong basin of A. Initially WB is equal to the weak basin of A (Line 2). In each iteration of Line 6, we take the current set WB, which is a subset of the weak basin of A, and remove from it all the states that have transitions to any state outside the current WB. These are the states from which there are paths to some other attractor $A' \neq A$ and hence they cannot be in the strong basin of A. Finally, when WB stabilises, we are left with the strong basin of A. A formal proof of correctness of Algorithm 1 can be found in the appendix of [17]. We shall use this algorithm in both the global minimal control algorithm and later in the decomposition-based algorithm.

We now use the algorithm COMPUTE_STRONG_BASIN to give a global algorithm, Algorithm 2, for the minimal simultaneous target control problem. Note that Algorithm 2 is worst-case exponential in the size of the input (the description of BN). Indeed, since the basin of attraction of A_t might well be equal to all the states of the entire transition system TS which is exponential in the description of BN. Now, although an efficient algorithm for this problem is highly unlikely, it is possible that when the network has a certain well-behaved structure, one can do better than this global approach. Most of the previous attempts at providing such an algorithm for such well-behaved networks either exploited exclusively the structure of the network or failed to minimise the number of driver nodes. Here we show that, when we take both the structure and the dynamics into account, we can have an algorithm which, for certain networks, is much more efficient than the global approach.

4 A DECOMPOSITION-BASED APPROACH

Note that our global solution for the minimal control problem, Algorithm 2, is generic, in that, we can plug into it any other algorithm for computing the basin of the target attractor and it would still work. Its performance, however, directly depends on the performance of the particular algorithm used to compute this basin.

In this section, we demonstrate an approach to compute the basin of attraction of A_t based on the decomposition of the BN into structural components called *blocks*. This will then be used to solve the minimal control problem. The approach is based on that of [13] for computing the attractors of asynchronous Boolean networks. The overall idea is as follows. The network is divided into *blocks* based on its strongly connected components. The blocks are then sorted topologically resulting in a dependency graph of the blocks which is a directed acyclic graph (DAG). The transition systems of the blocks are computed inductively in the sorted order and the target attractor A_t is then projected to these blocks. The local strong basins for each of these projections are computed in the transition system of the particular block. These local basins are then combined to compute the global basin $\mathsf{bas}(A_t)$.

4.1 Blocks

Let SCC denote the set of maximal strongly connected components (SCCs) of $\mathcal{G}_{\mathsf{BN}}$.[2] Let W be an SCC of $\mathcal{G}_{\mathsf{BN}}$. The set of parents of W is defined as $\mathsf{par}(W) = (\bigcup_{v \in W} \mathsf{par}(v)) \setminus W$.

Definition 4.1 (Basic Block). A basic block B is a subset of the vertices of V such that $B = W \cup \mathsf{par}(W)$ for some $W \in$ SCC.

Let $\mathcal{B}$ be the set of basic blocks of $\mathcal{G}_{\mathsf{BN}}$. Since every vertex of $\mathcal{G}_{\mathsf{BN}}$ is part of an SCC, we have $\bigcup \mathcal{B} = V$. The union of two or more basic blocks of $\mathcal{B}$ will also be called a *block*. For any block B, $|B|$ will denote the number of vertices in B. Using the set of basic blocks $\mathcal{B}$ as vertices, we can form a directed graph $\mathcal{G}_{\mathcal{B}} = (\mathcal{B}, E_{\mathcal{B}})$, which we shall call the *block graph* of BN. The vertices of $\mathcal{G}_{\mathcal{B}}$ are the basic blocks and for any pair of basic blocks $B', B \in \mathcal{B}, B' \neq B$, there is a directed edge from B' to B if and only if $B' \cap B \neq \emptyset$ and for every $v \in (B' \cap B)$, $\mathsf{par}(v) \cap B = \emptyset$. In such a case, B' is called a *parent* block of B and v is called a *control node* for B. Let $\mathsf{par}(B)$ and $\mathsf{ctr}(B)$ denote the set of parent blocks and the set of control nodes of B, respectively. It is easy to observe that

OBSERVATION 4. *$\mathcal{G}_{\mathcal{B}}$ is a directed acyclic graph (DAG).*

A block B (basic or non-basic) is called *elementary* if $\mathsf{par}(v) \subseteq B$ for every $v \in B$. B is called *non-elementary* otherwise. We shall henceforth assume that BN has k basic blocks and they are topologically sorted as $\{B_1, B_2, \ldots, B_k\}$. Note that for every $j : 1 \leq j \leq k$, $(\bigcup_{\ell=1}^{j} B_\ell)$ is an elementary block. We shall denote it as $\overline{B}_j$.

For two basic blocks B and B' where B is non-elementary, B' is said to be an ancestor of B if there is a path from B' to B in the block graph $\mathcal{G}_{\mathcal{B}}$. The ancestor-closure of a basic block B (elementary or non-elementary), denoted $\mathsf{ac}(B)$ is defined as the union of B with all its ancestors. Note that $\mathsf{ac}(B)$ is an elementary block and so is $\{\mathsf{ac}(B') \mid B' \in \mathsf{par}(B)\}$, which we denote as $\mathsf{ac}(B)^-$.

4.2 Projection of states and the cross operation

We shall assume that the vertices $\{v_1, v_2, \ldots, v_n\}$ of $\mathcal{G}_{\mathsf{BN}}$ inherit the ordering of the variables $\mathbf{x}$ of BN. Let B be a block of BN. Since B is a subset of V its state space is $\{0, 1\}^{|B|}$ and is denoted as S_B.

[2] By convention, we assume that a single vertex (with or without a self loop) is always an SCC, although it may not be maximal.

For any state $\mathbf{s} \in \mathsf{S}$, where $\mathbf{s} = (s_1, s_2, \ldots, s_n)$, the projection of $\mathbf{s}$ to B, denoted $\mathbf{s}|_B$ is the tuple obtained from $\mathbf{s}$ by suppressing the values of the variables not in B. Thus if $B = \{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\}$ then $\mathbf{s}|_B = (s_{i_1}, s_{i_2}, \ldots, s_{i_k})$. Clearly $\mathbf{s}|_B \in \mathsf{S}_B$. For a subset S' of S, $\mathsf{S}'|_B$ is defined as $\{\mathbf{s}|_B \mid \mathbf{s} \in \mathsf{S}'\}$.

Definition 4.2 (Cross Operation). Let B_1 and B_2 be two blocks of BN and let $\mathbf{s}_1$ and $\mathbf{s}_2$ be states of B_1 and B_2, respectively. $\mathbf{s}_1 \otimes \mathbf{s}_2$ is defined (called *crossable*) if there exists a state $\mathbf{s} \in \mathsf{S}_{B_1 \cup B_2}$ such that $\mathbf{s}|_{B_1} = \mathbf{s}_1$ and $\mathbf{s}|_{B_2} = \mathbf{s}_2$. $\mathbf{s}_1 \otimes \mathbf{s}_2$ is then defined to be this unique state $\mathbf{s}$. For any subsets S_1 of S_{B_1} and S_2 of S_{B_2}, $\mathsf{S}_1 \otimes \mathsf{S}_2$ is a subset of $\mathsf{S}_{B_1 \cup B_2}$ and is defined as:

$$\mathsf{S}_1 \otimes \mathsf{S}_2 = \{\mathbf{s}_1 \otimes \mathbf{s}_2 \mid \mathbf{s}_1 \in \mathsf{S}_1, \mathbf{s}_2 \in \mathsf{S}_2 \text{ and } \mathbf{s}_1 \text{ and } \mathbf{s}_2 \text{ are crossable}\}$$

Note that $\mathsf{S}_1 \otimes \mathsf{S}_2$ can be the empty set. The cross operation is easily seen to be associative. Hence for more than two states $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k$, $\mathbf{s}_1 \otimes \mathbf{s}_2 \otimes \ldots \mathbf{s}_k$ can be defined as $(((\mathbf{s}_1 \otimes \mathbf{s}_2) \otimes \ldots) \otimes \mathbf{s}_k)$. We have a similar definition for the cross operation on more than two sets of states.

Example 4.3. Let $\mathsf{BN} = (\mathbf{x}, \mathbf{f})$ be a Boolean network where $\mathbf{x} = (x_1, x_2, \ldots, x_5)$. Suppose BN has 2 blocks B_1 and B_2 with $B_1 = \{x_1, x_3, x_4\}$ and $B_2 = \{x_2, x_3, x_4, x_5\}$. Let $\mathbf{s} = (10011)$ be a state of BN. Then $\mathbf{s}|_{B_1} = (101)$, i.e. the 1st, 3rd and 4th components of $\mathbf{s}$ and $\mathbf{s}|_{B_2} = (0011)$, i.e. the 2nd, 3rd, 4th and 5th components of $\mathbf{s}$. Now, let $\mathbf{s}_1 = (001)$ be a state of B_1 and $\mathbf{s}_2 = (1010)$ be a state of B_2 then $\mathbf{s}_1 \otimes \mathbf{s}_2 = (01010)$ since this is the unique state of BN whose projections to B_1 and B_2 are $\mathbf{s}_1$ and $\mathbf{s}_2$, respectively.

4.3 Transition system of the blocks

The next step is to describe how to construct the 'local' transition systems of each of the blocks. These transition systems will be inductively defined starting from the elementary blocks and moving to the blocks further down the topological order. For an elementary block B (basic or non-basic), its transition system TS_B is given exactly as Definition 3.3 with the vertices being S_B. This is well-defined since by the definition of an elementary block, the update functions of the vertices of B do not depend on the value of any vertex outside B. On the other hand, the transition system of a non-elementary block B depends on the transitions of its parent blocks (or its control nodes in its parent blocks). The transition system of such a block thus has to be defined based on (some or all of) the transitions of its parent blocks.

Towards that let B be a non-elementary basic block of BN and let A be an attractor of the transition system of the elementary block $\mathrm{ac}(B)^-$ and let $\mathrm{bas}(A)$ be its (strong) basin of attraction. Then

Definition 4.4 (TS of non-elementary blocks). The transition system of B generated by $\mathrm{bas}(A)$ is defined as a tuple $\mathsf{TS}_B = (\mathsf{S}, \rightarrow)$ where the set of states S of TS_B is a subset of $\mathsf{S}_{\mathrm{ac}(B)}$ such that $\mathbf{s} \in \mathsf{S}$ if and only if $\mathbf{s}|_{\mathrm{ac}(B)^-} \in \mathrm{bas}(A)$ and for any two states $\mathbf{s}, \mathbf{s}' \in \mathsf{S}_{\mathrm{ac}(B)}$ there is a transition $\mathbf{s} \rightarrow \mathbf{s}'$ if and only if either $\mathrm{hd}(\mathbf{s}, \mathbf{s}') = 1$ and $\mathbf{s}'[i] = f_i(\mathbf{s})$ where $i = \arg(\mathrm{hd}(\mathbf{s}, \mathbf{s}'))$ or $\mathrm{hd}(\mathbf{s}, \mathbf{s}') = 0$ and there exists i such that $\mathbf{s}'[i] = f_i(\mathbf{s})$.

Remark. Our construction of the transition system of the non-elementary blocks is different from that used in [13]. There, for a non-elementary block B, the set of states of TS_B was a subset of S_B and the transitions for the control nodes of B were derived by projecting the transitions in the attractor of the parent block of B to these control nodes. It can be shown that such an approach does not work for the decomposition-based solution to the minimal simultaneous target-control problem that we aim for here and we need the full behaviour of the basin of the attractor of the parent blocks of B to generate the transition system of B.

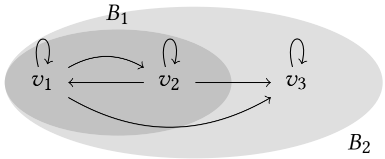


Figure 2: The blocks of BN.

4.4 The main results

We now give the key results of the above constructions which will form the basis of the decomposition-based control algorithm that we shall develop in the next section. The detailed proofs can be found in the appendix of [17].

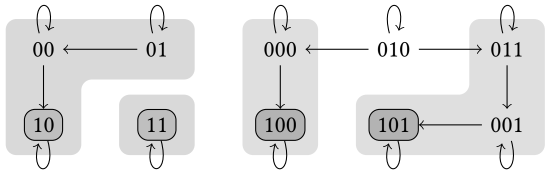
Suppose BN has k blocks which are topologically ordered as $\{B_1, B_2, \ldots, B_k\}$. Let TS be the transition system of BN and for every attractor A of TS and for every $j : 1 \leq j \leq k$ let $A_j = A|_{B_j}$ be the projection of A to B_j. We then have

Theorem 4.5 (Preservation of attractors). *Suppose for every attractor A of* TS *and for every $i : 1 \leq i < k$, if B_{i+1} is non-elementary then* TS_{i+1} *is realised by* $\mathrm{bas}(\otimes_{j \in I} A_j)$*, its basin w.r.t. the transition system for* $(\bigcup_{j \in I} B_j)$*, where I is the set of indices of the basic blocks in* $\mathrm{ac}(B_{i+1})^-$*. We then have, for every $i : 1 \leq i < k$, A_{i+1} is an attractor of* TS_{i+1}, $(\otimes_{j \in I} A_j \otimes A_{i+1})$ *is an attractor of the transition system for the elementary block* $(\bigcup_{j \in I} B_j \cup B_{i+1})$, $(\otimes_{j=1}^{i+1} A_j)$ *is an attractor of the transition system* $\overline{\mathsf{TS}}_{i+1}$ *of* $\overline{B}_{i+1}$ *and A is an attractor of* TS_k.

Theorem 4.6 (Preservation of basins). *Given the hypothesis and the notations of Theorem 4.5, we have* $(\otimes_{i \leq k} \mathrm{bas}(A_i)) = \mathrm{bas}(A)$ *where* $\mathrm{bas}(A)$ *is the basin of attraction of the attractor* $A = (A_1 \otimes A_2 \otimes \ldots \otimes A_k)$ *of* TS.

Example 4.7. Continuing with Example 3.6 and 3.9, we note that BN has two maximal SCCs $\{v_1, v_2\}$ and $\{v_3\}$. These give rise to two blocks $B_1 = \{v_1, v_2\}$ and $B_2 = \{v_1, v_2, v_3\}$ shown in Figure 2. B_1 is elementary whereas B_2 is non-elementary where B_1 is its parent and it has control nodes v_1 and v_2.

The transition system of block B_1 is shown in Figure 3(a). It has two attractors $\{(10)\}$ and $\{(11)\}$ shown in dark grey rectangles with their corresponding strong basins shown in grey regions of a lighter shade. The transition system of the block B_2 generated by the basin of the attractor $\{(10)\}$ of the block B_1 is shown in Figure 3(b). It has two attractors $\{(100)\}$ and $\{(101)\}$ shown again in dark grey rectangles with their corresponding basins of attractions shown in lighter grey. Note that, indeed, according to Theorem 4.5 we have that $\{(10)\} \otimes \{(100)\} = \{(100)\}$ and $\{(10)\} \otimes \{(101)\} = \{(101)\}$ are attractors of the global transition system of BN. Also note that taking the cross of the local basins of attractions does indeed result in the global basins.



(a) The TS of B_1, its attractors and basins.

(b) The TS of B_2 generated by the basin of the attractor $\{(10)\}$ of B_1.

Figure 3: The transition systems of the blocks B_1 and B_2.

4.5 The decomposition-based algorithm

Equipped with the results in Theorems 4.5 and 4.6, we can describe our procedure for computing the strong basin of the target attractor based on decomposing the BN into smaller blocks. We shall later use this procedure to give an algorithm for the minimal control problem. Towards that, Theorem 4.6 tells us that in order to compute bas(A_t) it is sufficient to compute the local basins of the projection of A_t to each block B_i (which by Theorem 4.5 is an attractor of B_i) and finally merge these local basins using the cross operation.

Algorithm 3 implements this idea in pseudo-code. It takes as input the graph $\mathcal{G}_{\mathrm{BN}}$ and the update functions **f** of a given Boolean network, and an attractor A and returns the strong basin of attraction of A. Line 2 decomposes $\mathcal{G}_{\mathrm{BN}}$ into the blocks $\mathcal{B}$ (resulting in k blocks) using the procedure Form_Block from [13] and line 3 topologically sorts the blocks by constructing the block graph $\mathcal{G}_{\mathcal{B}}$. Lines 5-7 decomposes the attractor A into its projection to the blocks. Lines 9-18 then cycles through the blocks of $\mathcal{B}$ in topological order and for each block B_i: if B_i is elementary then constructs its transition system TS$_i$ independently or, if B_i is non-elementary it constructs TS$_i$ realised by the basin of $(A_1 \otimes A_2 \otimes \ldots \otimes A_{i-1})$ which by Theorem 4.5 is an attractor of TS$_{i-1}$, the transition system for the elementary (non-basic) block $\overline{B}_{i-1}$. Thus at every iteration i of the for-loop the invariant that A_i is an attractor of TS$_i$ is maintained. The procedure Compute_Strong_Basin (lines 12,15), described in Algorithm 1, computes the strong basin of the attractor of TS$_i$ for the current block $\overline{B}_i$. Line 17 extends the global strong basin SB computed so far by crossing it with the local basin computed at each step. At the end of the for-loop SB will thus be equal to the global basin (by Theorem 4.6). It then easily follows that

Proposition 4.8. *Algorithm 3 correctly computes the strong basin of the attractor A.*

We now plug the procedure Compute_Strong_Basin_Decomp of Algorithm 3 into Algorithm 2 to derive our decomposition-based minimal target control algorithm, Algorithm 4, from source state **s** to target attractor A_t.

Algorithm 3 A decomposition-based procedure for the computation of strong basin

```
1:  procedure Compute_Strong_Basin_Decomp($\mathcal{G}_{BN}$,f,$A$)
2:      $\mathcal{B}$ := Form_Block($\mathcal{G}_{BN}$);
3:      $\mathcal{B}$ := Top_Sort($\mathcal{B}$);
4:      $k$ := size of $\mathcal{B}$; SB = $\phi$; SB$_i$ = $\emptyset$;            //for all i
5:      for $i$ = 1 to $k$ do
6:          $A_i$ :=Decompose($A, B_i$);      //Decompose the target
7:                                              attractor into block $B_i$
8:      end for
9:      for $i$ := 1 to $k$ do
10:         if $B_i$ is an elementary block then
11:             TS$_i$ := transition system of $B_i$;
12:             SB$_i$ :=Compute_Strong_Basin($\mathbf{f}|_{\overline{B}_i}, A_i$);
13:         else
14:             TS$_i$ := transition system of $B_i$ based on the basin of
        $(\otimes_{j<i}A_j)$ in TS$_{i-1}$;
15:             SB$_i$ :=Compute_Strong_Basin($\mathbf{f}|_{\overline{B}_i}, (\otimes_{j\leq i}A_j)$);
16:         end if
17:         SB =Cross (SB, SB$_i$);
18:     end for
19:     return SB
20: end procedure
```

Algorithm 4 Decomposition-based minimal simultaneous target control

```
procedure Decomp_Minimal_control($\mathcal{G}_{BN}$, f, s, $A_t$)
    Let SB = Compute_Strong_Basin_Decomp($\mathcal{G}_{BN}$, f, $A_t$)
    return arg(hd(s,SB))
end procedure
```

5 CASE STUDIES

To demonstrate the correctness and efficiency of our control framework, we compare our decomposition-based approach with the global approach on both real-life biological networks and randomly generated networks. Note that we do not compare our approach with the works by Mandon et al. [11, 12], as we are informed by the authors, through personal communication, that their current methods cannot deal with networks with more than around 20 nodes. The global approach and the decomposition-based approach, described by Algorithm 2 and Algorithm 4, are implemented in the software tool ASSA-PBN [14], which is based on the model checker [10] to encode BNs into the efficient data structure binary decision diagrams (BDDs). All the experiments are performed on a high-performance computing (HPC) platform, which contains CPUs of Intel Xeon X5675@3.07 GHz.

5.1 Real-life biological networks

The PC12 cell differentiation network was developed by Offermann et al. [16]. It is a comprehensive model used to clarify the cellular decisions towards proliferation or differentiation. It combines the temporal sequence of protein signalling, transcriptional response and subsequent autocrine feedback. The model shows the interactions between protein signalling, transcription factor activity and gene regulatory feedback in the regulation of PC12 cell differentiation after the stimulation of NGF. The PC12 cell network is simulated in the synchronous updating mode in [16], while we treat the network in the asynchronous mode, as per Definition 3.2. The PC12 cell network consists of 32 nodes and it has 7 single-state attractors. The network structure is divided into 19 blocks by our decomposition procedure (Form_Block in Algorithm 3).

Attractor	1		2		3		4		5		6		7	
	HD	#D	HD	#D	HD	#D	HD	#D	HD	#D	HD	#D	HD	#D
1	–	–	2	1	21	7	22	8	22	8	23	9	7	1
2	2	1	–	–	23	8	22	7	24	9	23	8	9	1
3	21	10	23	11	–	–	1	1	1	1	2	2	28	1
4	22	11	22	10	1	1	–	–	2	2	1	1	29	1
5	22	10	24	11	1	1	2	2	–	–	1	1	29	1
6	23	11	23	10	2	2	1	1	1	1	–	–	30	1
7	7	1	9	3	28	9	29	10	29	10	30	11	–	–

Table 1: The Hamming distance between attractors and the number of driver nodes computed for the PC12 cell network.

Attractor	1		2		3		4		5		6		7	
	HD	#D	HD	#D	HD	#D	HD	#D	HD	#D	HD	#D	HD	#D
9	1	1	6	2	3	2	8	3	21	2	26	3	23	3
10	6	2	1	1	8	3	3	2	26	3	21	2	28	4
11	3	2	8	3	1	1	6	2	23	3	28	4	21	2
12	8	3	3	2	6	2	1	1	28	4	23	3	26	3
13	10	2	15	3	12	3	17	4	12	1	17	2	14	2
14	15	3	10	2	17	4	12	3	17	2	12	1	19	3
15	12	3	17	4	10	2	15	3	14	2	19	3	12	1
16	17	4	12	3	15	3	10	2	19	3	14	2	17	2

Table 2: The Hamming distance between attractors and the number of driver nodes for the apoptosis network.

The apoptosis network was constructed by Schlatter et al. [18] based on extensive literature research. Apoptosis is a kind of programmed cell death, that has been linked to many diseases. In [18], Schlatter et al. took into consideration the survival and metabolic insulin pathways, the intrinsic and extrinsic apoptotic pathways and their crosstalks to build the Boolean network, which simulates apoptotic signal transduction pathways with regards to different input stimulus. The apoptosis network comprises 97 nodes and can be decomposed into 60 blocks. Using the asynchronous updating mode of BNs [Definition 3.2] 16 single-state attractors are detected when the housekeeping node is set to on and six nodes (FASL, FASL_2, IL_1,TNF, UV, UV_2) are set to off.

For the PC12 cell network and the apoptosis network, we aim to compute a minimal control C that can realise the minimal simultaneous single-step target control as explained in Section 3.4. That is to say, we compute the minimal set of driver nodes, whose simultaneous single-step control can drive the network from a source state to a target attractor. Since the attractors of the two networks are all single-state attractors, each of them can be taken as a source state. All possible combinations of source and target attractors of the networks are explored and each case is repeated 100 times.[3] The Hamming distances between attractors and the number of driver nodes are summarised in Table 1 and Table 2. The attractors are labelled with numbers. The numbers in the first column and the first row represent the source and target attractors, respectively. For each combination of source and target attractors, we list its Hamming distance (HD) and the number of driver nodes (#D). The numbers of driver nodes computed by the global and our decomposition-based approaches are identical, demonstrating the correctness of our decomposition-based approach. #D represents the results of both approaches.

Table 1 and Table 2 show that compared to the size of the network and the Hamming distance between the source and target attractors, the minimal set of driver nodes required is quite small. Especially for the apoptosis network with 97 nodes, the numbers of driver nodes are less than or equal to 4 for all the cases. The PC12 cell network always reaches the same steady state with "cell differentiation" set to on by setting NGF to 'on' [16]. To drive the network from any other attractor to this steady state, only NGF is required, which also shows the outstanding role of NGF in the network.

The speedups gained by our decomposition-based approach for different combinations of source and target attractors of the two networks are shown in Table 3 and Table 4. Each entity in the tables is an average value of the repeated experiments (100 times). The results show that our decomposition-based approach outperforms the global approach for any combination of source and target attractors. It is also obvious that the speedups are highly related to the target attractors, the speedups with different target attractors vary a lot regarding to the same source attractor.

5.2 Randomly generated networks

The same procedures are applied to three randomly generated Boolean networks with 100, 120 and 180 nodes. An overview of the three networks and their evaluation results is given in Table 6. The BNs with 100, 120 and 180 nodes are labelled as BN-100, BN-120 and BN-180 and they have 9, 4 and 2 single-state attractors, respectively. The global approach fails to compute the driver nodes for the BN-180 network and some cases of the BN-100 and BN-120 networks. The corresponding results are denoted as $*$.

Table 5 shows the time costs of the global approach and the decomposition-based approach on the BN-100 network. When the target attractors are 1, 6 and 8, the global approach fails to return any results within five hours. From Table 5, it is clear that the execution time is highly dependent on the target attractor. Especially for the

[3] Due to the space limit, here we only give parts of the results. More details on the networks and the evaluation results can be found in [17].

Attractor	Speedups						
	1	2	3	4	5	6	7
1	–	5.08	9.63	6.14	5.52	4.46	1.38
2	5.57	–	9.38	6.37	5.50	4.42	1.38
3	5.44	4.26	–	8.11	5.66	3.14	1.93
4	5.34	4.26	9.67	–	5.36	3.41	2.30
5	5.20	4.08	8.99	6.04	–	3.70	1.85
6	5.25	4.28	9.65	5.83	5.74	–	1.91
7	5.29	4.27	9.66	5.72	5.70	4.33	–

Table 3: Speedups achieved on the PC12 network.

Attractor	Speedups						
	1	2	3	4	5	6	7
9	2.06	2.00	2.26	2.22	20.27	17.44	24.88
10	2.13	1.93	2.37	2.14	22.13	17.58	24.95
11	2.14	1.94	2.37	2.15	22.22	17.24	24.71
12	2.17	1.98	2.38	2.18	22.11	17.12	25.31
13	2.17	1.96	2.40	2.18	22.58	17.65	24.86
14	2.15	1.95	2.40	2.17	22.30	17.46	25.09
15	2.14	1.95	2.39	2.17	22.24	18.09	24.85
16	2.16	1.97	2.42	2.19	22.58	17.53	25.23

Table 4: Speedups achieved on the apoptosis network.

global approach, it may cost a considerable amount of time when the basin of the target attractor is large. In terms of the number of driver nodes, the results computed by the two approaches are identical and they are not shown here due to the page limit.

Table 6 gives an overview of the networks and their evaluation results. Benefited from the fixpoint computation of strong basin, as described in Algorithm 1, both approaches are efficient. Compared with the global approach, our decomposition-based approach has an evident advantage in terms of efficiency, especially for large networks, thanks to its 'divide and conquer' strategy. The global approach fails to compute the results in some cases as it deals with the entire networks at once.

6 CONCLUSIONS AND FUTURE WORK

In this work, we have described a decomposition-based approach towards the computation of a minimal set of nodes (variables) to be simultaneously controlled of a BN so as to drive its dynamics from a source state to a target attractor. Our approach is generic and can be applied based on any algorithm for computing the strong basin of attraction of an attractor. For certain modular real-life networks, the approach results in significant increase in efficiency compared with a global approach and its generality means that the improvement in efficiency can be attained irrespective of the exact algorithm used for the computation of the strong basins.

We have only scratched the surface of an exciting approach towards the control of BNs which utilises both its structure and dynamics. We conclude by looking back critically at our approach, summarising various extensions and discussing future directions.

As mentioned in Section 1, the problem of minimal control is PSPACE-hard and efficient algorithms are unlikely for the general cases. Yet in retrospect, one might ask what is the inherent characteristic of our decomposition-based approach that makes it so efficient compared with the global approach for the real-life networks that we studied. We put forward a couple of heuristics which we believe explains and crucially determines the success of our approach. One such heuristic is that the basins of attraction computed at each step is small compared with the size of the transition system. This reduces the state space that needs to be considered in every subsequent step thus improving efficiency.

Another heuristic, which depends on the structure of the network, is that the number of blocks is small compared with the total number of nodes in the network. Otherwise, the approach has to compute a large number of local transition systems (as many as the number of blocks) which hampers its efficiency. However, the number of blocks in the network cannot be too few either. Otherwise, our approach comes close to the global approach in terms of efficiency. Note that if the entire network is one single giant block, then the decomposition-based approach is the same as the global approach (given that the same procedure is used for the computation of the strong basins) and there is no gain in efficiency. One might thus conjecture that there is an optimal block-to-node ratio, given which, our decomposition-based approach fares the best.

As discussed in Section 4.3, in [13] the construction of the TS of a non-elementary block B depends on the transitions of the control nodes of B which can be derived by projecting the transitions in the attractors of the parent block(s) of B to these control nodes. By this process of projection, the states of the TS of B had smaller dimension (equal to $|B|$) as compared with our current approach where the states of B have dimension equal to $|\mathrm{ac}(B)|$. This, in effect, can speed up the decomposition-based approach. Unfortunately, it turns out that such a projection does not work when we require to preserve the basins of the attractors across the blocks. Projection results in loss of information, without which it is not possible to derive the global basin of an attractor of the entire BN in terms of the cross of the local basins. However, it can be shown that if we do generate the transition system of a non-elementary block B by projecting the basins of attractions of the parent blocks to the control nodes of B, the cross of the local basins is a subset of the corresponding global basin of the attractor of the entire network. Thus, if we are ready to sacrifice accuracy for efficiency, such a projection-based technique might be faster for certain networks while not exactly giving the minimal nodes to control but a good-enough approximation of it. We would like to study the gain in efficiency in our approach by applying the above technique.

One way to reduce the number of 'small' blocks (which, as discussed, might degrade efficiency) might be to combine multiple basic blocks into larger blocks. While constructing the local transition systems, such merged blocks are treated as single basic blocks and their dynamics, attractors and basins are computed in one-go. We believe there are many real-life networks which might benefit from such a process of merging before applying our decomposition-based approach. This is another line of work that we are pursuing at the moment. As mentioned in the related work, the control approaches based on computation of the feedback vertex set [2, 15, 24] and the stable motifs [23] are promising approximate control algorithms for nonlinear dynamical networks. We would like to compare our approaches with these two in terms of efficiency and the number of driver nodes. Finally, we plan to extend our decomposition-based approach to the control of probabilistic Boolean networks [19, 21].

Attractors	Time (ms)								
	1	2	3	4	5	6	7	8	9
1	–	20,812	70,816	3,674	8,278	*	4182,340	*	3,935
	–	595	388	279	259	276	131	114	96
2	*	–	74,023	3,694	7,732	*	4072,920	*	3,808
	1,136	–	389	281	260	275	123	112	96
3	*	19,956	–	3,706	7,688	*	3750,570	*	3,865
	1,138	595	–	282	260	275	123	112	98
4	*	19,747	72,511	–	7,721	*	2021,220	*	3,904
	1,138	595	388	–	260	276	123	112	97
5	*	22,598	29,760	3,275	–	*	4424,750	*	4,249
	1,137	595	389	279	–	275	123	112	94
6	*	19,744	73,355	3,707	7,750	–	2149,410	*	3,883
	1,141	595	389	282	259	–	121	111	96
7	*	19,742	72,197	3,706	7,689	*	–	*	3,886
	1,139	595	390	280	259	274	–	111	97
8	*	19,763	73,115	3,706	7,701	*	2089,800	–	3,842
	1,139	594	388	281	259	274	124	–	96
9	*	19,719	73,397	3,710	7,460	*	2343,120	*	–
	1,139	595	389	283	259	274	124	111	–

Table 5: Time costs of the global approach and the decomposition-based approach on the BN-100 network.

Networks	#	#	#	Range of	Range of	Range of
	nodes	blocks	attractors	t_{global} (ms)	t_{decom} (ms)	speedups
PC12	32	19	7	16 – 56	5 – 12	1.375 – 9.672
apoptosis	97	60	16	1,472 – 46,560	747 – 994	1.932 – 51.504
BN-100	100	36	9	3,275 – 4424,750	94 – 1,141	11.738 – 35973.577
BN-120	120	27	4	257,3 – 14774,2	2,840 – 6,466	39,89 – 4818,72
BN-180	180	62	2	*	1,402 – 1,462	*

Table 6: An overview of the evaluation results.

ACKNOWLEDGMENTS

This work was partially supported by the project SEC-PBN funded by University of Luxembourg and the ANR-FNR project AlgoReCell.

REFERENCES

[1] E. Czeizler, C. Gratie, W. K. Chiu, K. Kanhaiya, and I. Petre. 2016. Target Controllability of Linear Networks. In *Proc. 14th Conference on Computational Methods in Systems Biology (LNCS)*, Vol. 9859. Springer, 67–81.

[2] B. Fiedler, A. Mochizuki, G. Kurosawa, and D. Saito. 2013. Dynamics and control at feedback vertex sets. I: Informative and determining nodes in regulatory networks. *Journal of Dynamics and Differential Equations* 25, 3 (2013), 563–604.

[3] J. Gao, Y.-Y. Liu, R. M. D'Souza, and A.-L. Barabási. 2014. Target control of complex networks. *Nature Communications* 5 (2014), 5415.

[4] A. J. Gates and L. M. Rocha. 2016. Control of complex networks requires both structure and dynamics. *Scientific Reports* 6 (2016), 24456.

[5] T. Graf and T. Enver. 2009. Forcing cells to change lineages. *Nature* 462, 7273 (2009), 587–594.

[6] S. Huang. 2001. Genomics, complexity and drug discovery: insights from Boolean network models of cellular regulation. *Pharmacogenomics* 2, 3 (2001), 203–222.

[7] S. A. Kauffman. 1969. Homeostasis and differentiation in random genetic control networks. *Nature* 224 (1969), 177–178.

[8] Y.-C. Lai. 2014. Controlling complex, non-linear dynamical networks. *National Science Review* 1, 3 (2014), 339–341.

[9] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási. 2011. Controllability of complex networks. *Nature* 473 (2011), 167–73.

[10] A. Lomuscio, H. Qu, and F. Raimondi. 2017. MCMAS: An open-source model checker for the verification of multi-agent systems. *STTT* 19, 1 (2017), 9–30.

[11] H. Mandon, S. Haar, and L. Paulevé. 2016. Relationship between the reprogramming determinants of Boolean networks and their interaction graph. In *Proc. 5th Workshop on Hybrid Systems Biology (LNCS)*, Vol. 9957. Springer, 113–127.

[12] H. Mandon, S. Haar, and L. Paulevé. 2017. Temporal reprogramming of Boolean networks. In *Proc. 15th Conference on Computational Methods in Systems Biology (LNCS)*, Vol. 10545. Springer, 179–195.

[13] A. Mizera, J. Pang, H. Qu, and Q. Yuan. 2018. Taming Asynchrony for Attractor Detection in Large Boolean Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (Special issue of APBC'18)* (2018).

[14] A. Mizera, J. Pang, and Q. Yuan. 2016. ASSA-PBN 2.0: A software tool for probabilistic Boolean networks. In *Proc. 14th International Conference on Computational Methods in Systems Biology (LNCS)*, Vol. 9859. Springer, 309–315.

[15] A. Mochizuki, B. Fiedler, G. Kurosawa, and D. Saito. 2013. Dynamics and control at feedback vertex sets. II: A faithful monitor to determine the diversity of molecular activities in regulatory networks. *J. Theoretical Biology* 335 (2013), 130–146.

[16] B. Offermann, S. Knauer, A. Singh, M. L. Fernández-Cachón, M. Klose, S. Kowar, H. Busch, and M. Boerries. 2016. Boolean modeling reveals the necessity of transcriptional regulation for bistability in PC12 cell differentiation. *Frontiers in Genetics* 7 (2016), 44.

[17] S. Paul, C. Su, J. Pang, and A. Mizera. 2018. A decomposition-based approach towards the control of Boolean networks (technical report). (2018). Available online at http://arxiv.org/abs/1804.07221.

[18] R. Schlatter, K. Schmich, I. A. Vizcarra, P. Scheurich, T. Sauter, C. Borner, M. Ederer, I. Merfort, and O. Sawodny. 2009. ON/OFF and Beyond - A Boolean Model of Apoptosis. *PLOS Computational Biology* 5, 12 (2009), e1000595.

[19] I. Shmulevich and E. R. Dougherty. 2010. *Probabilistic Boolean Networks: The Modeling and Control of Gene Regulatory Networks.* SIAM Press.

[20] A. del Sol and N.J. Buckley. 2014. Concise review: A population shift view of cellular reprogramming. *STEM CELLS* 32, 6 (2014), 1367–1372.

[21] P. Trairatphisan, A. Mizera, J. Pang, A.-A. Tantar, J. Schneider, and T. Sauter. 2013. Recent development and biomedical applications of probabilistic Boolean networks. *Cell Communication and Signaling* 11 (2013), 46.

[22] L.-Z. Wang, R.-Q. Su, Z.-G. Huang, X. Wang, W.-X. Wang, C. Grebogi, and Y.-C. Lai. 2016. A geometrical approach to control and controllability of nonlinear dynamical networks. *Nature Communications* 7 (2016), 11323.

[23] J. G. T. Zañudo and R. Albert. 2015. Cell fate reprogramming by control of intracellular network dynamics. *PLoS Computational Biology* 11, 4 (2015), e1004193.

[24] J. G. T. Zañudo, G. Yang, and R. Albert. 2017. Structure-based control of complex networks with nonlinear dynamics. *PNAS* 114, 28 (2017), 7234–7239.