This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2019.2915081, IEEE/ACM Transactions on Computational Biology and Bioinformatics

An efficient approach towards the source-target control of Boolean networks

Soumya Paul, Cui Su, Jun Pang, Andrzej Mizera

Abstract—We study the problem of computing a minimal subset of nodes of a given asynchronous Boolean network that need to be perturbed in a single-step to drive its dynamics from an initial state to a target steady state (or *attractor*), which we call the *source-target control* of Boolean networks. 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 this view, we derive a decomposition-based solution to the minimal source-target control problem which can be significantly faster than the existing approaches on large networks. We then show that the solution can be further optimised if we take into account appropriate information about the source state. We apply our solutions to both real-life biological networks and randomly generated networks, demonstrating the efficiency and efficacy of our approach.

Index Terms—Boolean networks, attractors, network control, decomposition

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 [1], [2]. 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], [4], [5].

Boolean networks (BNs), first introduced by Kauffman [6], 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 [6] and also correspond to functional cellular states such as proliferation, apoptosis differentiation etc. [7].

Cellular reprogramming, or the control of the GRNs and their signalling pathways therefore amount to being able to drive the dynamics of the associated BN from an attractor to another 'desirable' target attractor by perturbing 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 perturbed or the perturbation 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 source-target control problem) is computationally difficult [8], [9] and does not scale well to large networks. Thus a simple global approach (see Section 3.5 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 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.

• *S. Paul is with the Computer Science and Communications Research Unit, University of Luxembourg. C. Su is with the Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg. J. Pang is with the Computer Science and Communications Research Unit and the Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg. A. Mizera is with the Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland.*
E-mail: soumya.paul@uni.lu, cui.su@uni.lu, jun.pang@uni.lu, andrzej.mizera@gmail.com

Manuscript received April 19, 2005; revised August 26, 2015.

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

- The problem of computing the minimal set of nodes to be perturbed in a single time-step (hence 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).
- The network structure of a large BN can be explored to decompose it into smaller *blocks*. The strong basins of attractions of the projections 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 (with slight modifications) 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 (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.
- Furthermore, taking relevant information about source state $\mathbf{s}$ into consideration, we can avoid the computation of local strong basins in some blocks. This results in a subset of the global strong basin of A_t, which still contains all the states that are required for the computation of the minimal control. Such an optimisation can accelerate the computation of the basin of A_t.
- We have implemented our decomposition-based approach (Algorithm 4) and its optimisation (Algorithm 5) using Algorithm 1b (which is a slight modification of Algorithm 1) as the basis for computation of strong basins, and applied them 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. Moreover, its efficiency is further improved by the optimisation of Algorithm 5.

2 Related Work

In recent years, several approaches have been developed for the control of complex networks [3], [4], [8], [9], [10], [11], [12], [13], [14], [15]. Among them, the methods [3], [4], [13] were proposed to tackle the control of networks with linear time-invariant dynamics. Liu et al. [3] 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 [4]. 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. [13] 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 [4] 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 [8], [9], [10], [11], [12], [14], [15] are designed for networks governed by non-linear dynamics. Among these methods, the ones based on the computation of the feedback vertex set (FVS) [10], [11], [15] and the 'stable motifs' of the network [12] 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 [14], 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. [8], [9] proposed approaches towards the control of asynchronous BNs. In particular, in [8] 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 [9]. 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 [9]. 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. [16] 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 not be sufficient.

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

Note that there is the related and extensively studied framework of Boolean control networks (BCNs) in the literature which is used to investigate the controllability and stability of Boolean networks using external inputs [17], [18], [19], [20]. These inputs are applied to a fixed subset of variables of the given BN for a finite sequence of steps. The controllability of the BN then boils down to computing sequences of values for these inputs for the required number of steps to drive the BN to a desired state. Although the control method we study in this work can be cast into the setting of BCNs, we do not explore that direction here and shall henceforth refrain from the use or mention of BCNs. Nevertheless, we would definitely like to explore the connection between BCNs and our methods of control in the near future.

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 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 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 $\mathsf{par}(v) = \{v' \in V \mid v' \rightarrow v\}$. 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 $\mathbf{S}$ 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 $\mathbf{s}, \mathbf{s}' \in \mathbf{S}$, the Hamming distance between $\mathbf{s}$ and $\mathbf{s}'$ will be denoted as $\mathsf{hd}(\mathbf{s}, \mathbf{s}')$ and $\arg(\mathsf{hd}(\mathbf{s}, \mathbf{s}')) \subseteq \{1, 2, \ldots, n\}$ will denote the set of indices in which $\mathbf{s}$ and $\mathbf{s}'$ differ. For a state $\mathbf{s}$ and a subset $\mathbf{S}' \subseteq \mathbf{S}$, the Hamming distance between $\mathbf{s}$ and $\mathbf{S}'$ is defined as the minimum of the Hamming distances between $\mathbf{s}$ and all the states in $\mathbf{S}'$. That is, $\mathsf{hd}(\mathbf{s}, \mathbf{S}') = \min_{\mathbf{s}' \in \mathbf{S}'} \mathsf{hd}(\mathbf{s}, \mathbf{s}')$. We let $\arg(\mathsf{hd}(\mathbf{s}, \mathbf{S}'))$ denote the set of subsets of $\{1, 2, \ldots, n\}$ such that $I \in \arg(\mathsf{hd}(\mathbf{s}, \mathbf{S}'))$ 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 2 (Asynchronous dynamics of Boolean networks). Suppose $\mathbf{s}_0 \in \mathbf{S}$ is an initial state of BN. The asynchronous evolution of BN is a function $\xi : \mathbb{N} \rightarrow \wp(\mathbf{S})$ such that $\xi(0) = \mathbf{s}_0$ and for every $j \geq 0$, if $\mathbf{s} \in \xi(j)$ then $\mathbf{s}' \in \xi(j+1)$ is a possible *next state* if and only if either $\mathsf{hd}(\mathbf{s}, \mathbf{s}') = 1$ and $\mathbf{s}'[i] = f_i(\mathbf{s})$ where $i = \arg(\mathsf{hd}(\mathbf{s}, \mathbf{s}'))$ or $\mathsf{hd}(\mathbf{s}, \mathbf{s}') = 0$ 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 (Transition system of BN). The transition system of BN, denoted by the generic notation TS is a tuple $(\mathbf{S}, \rightarrow)$ where the vertices are the set of states $\mathbf{S}$ and for any two states $\mathbf{s}$ and $\mathbf{s}'$ there is a directed edge from $\mathbf{s}$ to $\mathbf{s}'$, denoted $\mathbf{s} \rightarrow \mathbf{s}'$ iff $\mathbf{s}'$ is a possible next state according to the asynchronous evolution function ξ of BN.

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 $\mathbf{S}'$ of $\mathbf{S}$ is a path from $\mathbf{s}$ to any state $\mathbf{s}' \in \mathbf{S}'$. For any state $\mathbf{s} \in \mathbf{S}$, let $\mathsf{pre}_{\mathsf{TS}}(\mathbf{s}) = \{\mathbf{s}' \in \mathbf{S} \mid \mathbf{s}' \rightarrow \mathbf{s}\}$ and let $\mathsf{post}_{\mathsf{TS}}(\mathbf{s}) = \{\mathbf{s}' \in \mathbf{S} \mid \mathbf{s} \rightarrow \mathbf{s}'\}$. $\mathsf{pre}_{\mathsf{TS}}(\mathbf{s})$ contains all the states that can reach $\mathbf{s}$ by performing a single transition in TS and $\mathsf{post}_{\mathsf{TS}}(s)$ contains all the states that can be reached from $\mathbf{s}$ by a single transition in TS. $\mathsf{pre}_{\mathsf{TS}}(\mathbf{s})$ and $\mathsf{post}_{\mathsf{TS}}(\mathbf{s})$ are often called the set of *predecessors* and *successors* of $\mathbf{s}$. Note that, by definition, $\mathsf{hd}(\mathbf{s}, \mathsf{pre}_{\mathsf{TS}}(\mathbf{s})) \leq 1$ and $\mathsf{hd}(\mathbf{s}, \mathsf{post}_{\mathsf{TS}}(\mathbf{s})) \leq 1$. $\mathsf{pre}_{\mathsf{TS}}$ and $\mathsf{post}_{\mathsf{TS}}$ can be lifted to a subset $\mathbf{S}'$ of $\mathbf{S}$ as: $\mathsf{pre}_{\mathsf{TS}}(\mathbf{S}') = \bigcup_{\mathbf{s} \in \mathbf{S}'} \mathsf{pre}_{\mathsf{TS}}(\mathbf{s})$ and $\mathsf{post}_{\mathsf{TS}}(\mathbf{S}') = \bigcup_{\mathbf{s} \in \mathbf{S}'} \mathsf{post}_{\mathsf{TS}}(\mathbf{s})$. We define $\mathsf{pre}^{i+1}_{\mathsf{TS}}(\mathbf{S}') = \mathsf{pre}_{\mathsf{TS}}(\mathsf{pre}^{i}_{\mathsf{TS}}(\mathbf{S}'))$ and $\mathsf{post}^{i+1}_{\mathsf{TS}}(\mathbf{S}') = \mathsf{post}_{\mathsf{TS}}(\mathsf{post}^{i}_{\mathsf{TS}}(\mathbf{S}'))$ where $\mathsf{pre}^{0}_{\mathsf{TS}}(\mathbf{S}') = \mathsf{post}^{0}_{\mathsf{TS}}(\mathbf{S}') = \mathbf{S}'$. For a state $\mathbf{s} \in \mathbf{S}$, $\mathsf{reach}_{\mathsf{TS}}(\mathbf{s})$ 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 fixpoint of the successor operation which is often denoted as $\mathsf{post}^{*}_{\mathsf{TS}}$. Thus, $\mathsf{reach}_{\mathsf{TS}}(\mathbf{s}) = \mathsf{post}^{*}_{\mathsf{TS}}(\mathbf{s})$.

Definition 4 (Attractor). An attractor A of TS (or of BN) is a minimal subset of states of $\mathbf{S}$ such that for every $\mathbf{s} \in A$, $\mathsf{reach}_{\mathsf{TS}}(\mathbf{s}) = A$.

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 $\mathsf{reach}_{\mathsf{TS}}(\mathbf{s}) \cap A \neq \emptyset$. Attractors represent the stable behaviour of the BN according to the dynamics. The network starting at any initial state $\mathbf{s}_0 \in \mathbf{S}$ 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.

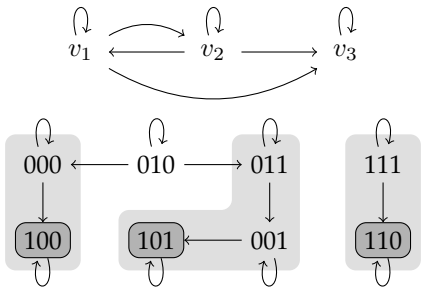


Fig. 1: The graph of BN and its transition system.

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

Definition 5 (Basin). Let A be an attractor of TS.

- **Weak basin:** The weak basin of attraction of A with respect to TS, is defined as $\mathsf{bas}^W_{\mathsf{TS}}(A) = \{\mathbf{s} \in \mathbf{S} \mid \mathsf{reach}_{\mathsf{TS}}(\mathbf{s}) \cap A \neq \emptyset\}$ which equals the fixpoint of the predecessor opration on A and is often denoted as $\mathsf{pre}^*_{\mathsf{TS}}$. Thus, $\mathsf{bas}^W_{\mathsf{TS}}(A) = \mathsf{pre}^*_{\mathsf{TS}}(A)$.
- **Strong basin:** The strong basin of attraction of A with respect to TS, is defined as $\mathsf{bas}^S_{\mathsf{TS}}(A) = \mathsf{bas}^W_{\mathsf{TS}}(A) \setminus (\bigcup_{A'} \mathsf{bas}^W_{\mathsf{TS}}(A'))$ where the union is over all attractors A' of TS such that $A' \neq A$.

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 $\mathbf{s} \in \mathsf{bas}^S_{\mathsf{TS}}(A)$ then $\mathbf{s} \notin \mathsf{bas}^W_{\mathsf{TS}}(A')$ for any other attractor A'. We need the notion of strong basin to *ensure* reachability to the target attractor after applying perturbations.

Example 1. 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 Fig. 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.

Henceforth, to avoid clutter, we shall drop the subscript TS when the transition system is clear from the context. Also, we shall often drop the superscript S when dealing with strong basins.

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, perturbing the values of the variables of the network. Such a perturbation 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 perturbed (a) *simultaneously* – the perturbation is applied to all the variables at once or (b) *sequentially* – the perturbation 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 source-target control as follows.

Definition 6 (Simultaneous control). A simultaneous control C is a (possibly empty) subset of $\{1, 2, \ldots, n\}$. For a state $\mathbf{s} \in \mathbf{S}$, the application of C to $\mathbf{s}$, denoted $\mathsf{C}(\mathbf{s})$ is defined as the state $\mathbf{s}' \in \mathbf{S}$ such that $\mathbf{s}'[i] = (1 - \mathbf{s}[i])$ if $i \in \mathsf{C}$ and $\mathbf{s}'[i] = \mathbf{s}[i]$ otherwise. Given a control C, the set of vertices $\{v_i \mid i \in \mathsf{C}\}$ of $\mathcal{G}_{\mathsf{BN}}$ 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 are the perturbations applied for just a single time step, they are also applied to as few of the nodes of the Boolean network as possible. The minimal simultaneous single-step source-target control problem for Boolean networks that we are thus interested in can be formally stated as follows.

Minimal simultaneous source-target control: Given a Boolean network BN, a 'source state' $\mathbf{s} \in \mathbf{S}$ and a 'target attractor' A_t of TS, compute a simultaneous control C such that after the application of C to $\mathbf{s}$, 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 source-target control (STC)* from $\mathbf{s}$ to A_t. The set of all minimal control from $\mathbf{s}$ to A_t will be denoted as $\mathbb{C}^{\mathbf{s} \to A_t}_{\min}$.

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. 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 (the driver nodes) so that after applying the toggle, the dynamics eventually and surely reaches A_t. It is known that in general the problem is computationally difficult – PSPACE-hard [8] and unless certain open conjectures in computational complexity are false, solving it 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 2. It is important to note that if the BN is in some state $\mathbf{s} \in \mathsf{bas}(A)$ in some time step t, that is if $\xi(t) = \mathbf{s}$, and assuming that every variable is updated infinitely often (faireness assumption), then by the definition of $\mathsf{bas}(A)$, 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, $\mathbb{C}_{\min}^{\mathbf{s}\to A_t}$ can easily be seen to be equal to $\arg(\mathsf{hd}(\mathbf{s}, \mathsf{bas}(A_t)))$. In other words

Proposition 1. A control C from $\mathbf{s}$ to A_t is minimal if and only if $\mathsf{C}(\mathbf{s}) \in \mathsf{bas}(A_t)$ and $\mathsf{C} \in \arg(\mathsf{hd}(\mathbf{s}, \mathsf{bas}(A)))$.

Proof. If $\mathsf{C}(\mathbf{s}) \notin \mathsf{bas}(A_t)$ then either (a) $\mathsf{C}(\mathbf{s}) \notin \mathsf{bas}^W_{\mathsf{TS}}(A_t)$ or (b) $\mathsf{C}(\mathbf{s}) \in \mathsf{bas}^W_{\mathsf{TS}}(A_t)$. If (a) holds, then there is no path from $\mathsf{C}(\mathbf{s})$ to A_t and if (b) holds, then there is a path from $\mathsf{C}(\mathbf{s})$ 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 $\mathsf{C} \notin \arg(\mathsf{hd}(\mathbf{s}, \mathsf{bas}(A)))$, 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 2. Continuing with Example 1, suppose we are in source state $\mathbf{s} = (101)$ (which is also an attractor) and we want to apply (minimal simultaneous) STC to $\mathbf{s}$ so that 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 $\mathsf{C} = \{2, 3\}$. However, if we note that the state (111) is in $\mathsf{bas}(A_t)$, we can simply apply the STC $\mathsf{C}' = \{2\}$ and the dynamics of the BN will ensure that it eventually reaches A_t. Indeed, C' is also the minimal STC in this case.

3.5 A global approach

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 STC problem based on a global computation of the basin of the target attractor A_t. A slightly modified version of the same algorithm will act as a reference for comparing the decomposition-based algorithm which we shall later develop.

3.5.1 Computation of basins

We first introduce procedures COMP_WB and COMP_SB, described in Algorithm 1, for the computation of the weak basin and strong basin of an attractor A based on fixpoint approaches. Procedure COMP_WB computes the weak basin of A by computing the fixpoint $\mathsf{pre}^*_{\mathsf{TS}}(A)$ (as per Definition 5). The most important step of COMP_SB is line 13, 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 13, 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. Below, we give a formal proof of the correctness of Algorithm 1. We shall use the procedure COMP_SB for the global minimal control algorithm and later, a minor modification of the procedure, which we call COMP_SB_REL, for the decomposition-based algorithm.

Algorithm 1 Computation of weak and strong basins

```
1: procedure COMP_WB((f, A))// Compute the weak basin
2:     Initialise WB' = ∅, WB = A;
3:     while WB ≠ WB' do
4:         WB'= WB
5:         WB = pre(WB')
6:     end while
7:     return WB;
8: end procedure
9: procedure COMP_SB(f,A)   // Compute the strong basin
10:    WB := COMP_WB(f, A);
11:    Initialise SB = ∅;
12:    while SB ≠ WB do
13:            If SB≠ ∅ do WB := SB;
14:            SB:=WB\(pre(post(WB) \ WB)∩WB);
15:    end while
16:    return SB;
17: end procedure
```

3.5.2 Correctness of Algorithm 1

The correctness of the procedure COMP_WB is straightfoward and follows from the fact that $\mathsf{bas}^W(A) = \mathsf{pre}^*(A)$ (Definition 5). Below we show the correctness of the procedure COMP_SB. Based on line 13 of Algorithm 1, we define an operator F on $\mathbf{S}$ as follows. For any subset $\mathbf{T}$ of state:

$$F(\mathbf{T}) = \mathbf{T} \setminus (\mathsf{pre}(\mathsf{post}(\mathbf{T}) \setminus \mathbf{T}) \cap \mathbf{T})$$

It is easy to see that F is monotonically decreasing and hence its greatest fixed point, F^∞, exists. Thus, to prove the correctness of Algorithm 1, it is enough to show that for any attractor A of TS, $F^\infty(\mathsf{bas}^W(A)) = \mathsf{bas}^S(A)$. That is, to compute the strong basin of A once can start with its weak basin and apply the operator F repeatedly till a fixed point is reached which gives its strong basin. The operation has to be repeated m times where m is the index of $F^\infty(\mathsf{bas}^W(A))$. We start by proving the following lemmas.

Lemma 1. For any state $\mathbf{s} \in \mathbf{S}$, if $\mathbf{s} \notin \mathsf{bas}^S(A)$ then $\mathbf{s} \notin F^\infty(\mathsf{bas}^W(A))$.

Proof. Suppose for some $\mathbf{s} \in \mathbf{S}$, $\mathbf{s} \notin \mathsf{bas}^S(A)$. Then either (i) there is no path from $\mathbf{s}$ to A or (ii) there is a path from $\mathbf{s}$ to another attractor $A' \neq A$ of TS. If (i) holds then $\mathbf{s} \notin \mathsf{bas}^W(A)$ either and hence $\mathbf{s} \notin F^\infty(\mathsf{bas}^W(A))$. So suppose (ii) holds and there is a path from $\mathbf{s}$ to another attractor $A' \neq A$. Consider the shortest such path $\mathbf{s}_0 \to \mathbf{s}_1 \to \ldots \to \mathbf{s}_n$, where $\mathbf{s}_0 = \mathbf{s}$ and $\mathbf{s}_n \in A'$ and let $\mathbf{s}_i \to \mathbf{s}_{(i+1)}$, $0 \leq i < n$ be the first transition along this path that moves out of $\mathsf{bas}^W(A)$. That is, $\mathbf{s}_i \in \mathsf{bas}^W(A)$ but $\mathbf{s}_{(i+1)} \notin \mathsf{bas}^W(A)$. We claim that $\mathbf{s} \notin F^j(\mathsf{bas}^W(A))$ for all $j \geq (i+1)$. That is, $\mathbf{s}$ is removed in the $(i+1)$th step in the inductive construction of $F^\infty(\mathsf{bas}^W(A))$. We prove this by induction on i.

Suppose $i = 0$. Then there is already a transition from $\mathbf{s}$ out of $\mathsf{bas}^W(A)$ and hence $\mathbf{s} \in (\mathsf{pre}(\mathsf{post}(\mathsf{bas}^W(A)) \setminus A) \cap \mathsf{bas}^W(A))$. Thus $\mathbf{s} \notin F(\mathsf{bas}^W(A))$. Next, suppose $i > 0$ and the premise holds for all $j : 0 \leq j < i$. Then by induction hypothesis we have $\mathbf{s}_1 \notin F^i(\mathsf{bas}^W(A))$. Hence $\mathbf{s} \in (\mathsf{pre}(\mathsf{post}(F^i(\mathsf{bas}^W(A))) \setminus F^i(\mathsf{bas}^W(A))) \cap F^i(\mathsf{bas}^W(A)))$.

and will be removed in the $(i+1)$th step of the inductive construction. □

For the converse direction, first, we easily observe from the definition of weak and strong basins that:

Lemma 2. Let A be an attractor of TS. Then

- $\mathsf{bas}^S(A) \subseteq \mathsf{bas}^W(A)$,
- for any state $\mathbf{s} \in \mathbf{S}$, $\mathbf{s} \in \mathsf{bas}^S(A)$ iff, for all transitions $\mathbf{s} \to \mathbf{s}'$, we have $\mathbf{s}' \in \mathsf{bas}^S(A)$.

We thus have

Lemma 3. For any state $\mathbf{s} \in \mathbf{S}$, if $\mathbf{s} \notin F^\infty(\mathsf{bas}^W(A))$ then $\mathbf{s} \notin \mathsf{bas}^S(A)$.

Proof. For some state $\mathbf{s} \in \mathbf{S}$, if $\mathbf{s} \notin F^\infty(\mathsf{bas}^W(A))$ then either $\mathbf{s} \notin \mathsf{bas}^W(A)$, in which case $\mathbf{s} \notin \mathsf{bas}^S(A)$ [by Lemma 2] or $\mathbf{s} \in \mathsf{bas}^W(A)$ but gets removed from $F^\infty(\mathsf{bas}^W(A))$ at the ith step of the inductive construction for some $i \geq 1$. We do an induction on i to show that in that case $\mathbf{s} \notin \mathsf{bas}^S(A)$. Suppose $i = 1$. Then by definition $\mathbf{s} \in (\mathsf{pre}(\mathsf{post}(\mathsf{bas}^W(A)) \setminus \mathsf{bas}^W(A)) \cap \mathsf{bas}^W(A))$ which means there is a transition from $\mathbf{s}$ to some $\mathbf{s}' \notin \mathsf{bas}^W(A)$. Thus $\mathbf{s} \notin \mathsf{bas}^S(A)$ [by Lemma 2]. Next suppose $i > 1$ and the premise holds for all $j : 1 \leq j < i$. Then, $\mathbf{s} \in (\mathsf{pre}(\mathsf{post}(F^{(i-1)}(\mathsf{bas}^W(A))) \setminus F^{(i-1)}(\mathsf{bas}^W(A))) \cap F^{(i-1)}(\mathsf{bas}^W(A)))$. This means there is a state $\mathbf{s}' \in F^{(i-1)}(\mathsf{bas}^W(A))$ such that there is a transition from $\mathbf{s}$ to $\mathbf{s}'$. But since by induction hypothesis $\mathbf{s}' \notin \mathsf{bas}^S(A)$ we must have that $\mathbf{s} \notin \mathsf{bas}^S(A)$ [by Lemma 2]. □

Combining Lemmas 1 and 3 we have

Theorem 1 (Correctness of Algorithm 1). For any attractor A of TS we have, $\mathsf{bas}^S(A) = F^\infty(\mathsf{bas}^W(A))$.

3.5.3 The global algorithm

We now use the algorithm COMP_SB to give a global algorithm, Algorithm 2, for the minimal simultaneous STC 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. 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.

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.

Algorithm 2 Global minimal simultaneous target control

```
1: procedure GLOBAL_MINIMAL_CONTROL(f, s, A_t)
2:     SB := COMP_SB(f, A_t);
3:     return arg(hd(s,SB));
4: end procedure
```

4 A DECOMPOSITION-BASED APPROACH

In this section, we demonstrate an approach to compute the strong 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 STC problem. The approach is based on that of [21] 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 (SCCs). 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 strong basins are then combined to compute the global strong basin $\mathsf{bas}(A_t)$.

4.1 Blocks

Let SCC denote the set of maximal 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 7 (Basic Block). A basic block B is a subset of V such that $B = W \cup \mathsf{par}(W)$ for some $W \in \mathsf{SCC}$.

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 3. $\mathcal{G}_{\mathcal{B}}$ 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 all the ancestors

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.

of B. 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 $\mathbf{S}_B$. For any state $\mathbf{s} \in \mathbf{S}$, 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 $\mathbf{s}|_B \in \mathbf{S}_B$. For a subset $\mathbf{S}'$ of $\mathbf{S}$, $\mathbf{S}'|_B$ is defined as $\{\mathbf{s}|_B \mid \mathbf{s} \in \mathbf{S}'\}$.

Definition 8 (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 $\mathbf{s} \in \mathbf{S}_{B_1 \cup B_2}$ 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 $\mathbf{S}_1$ of $\mathbf{S}_{B_1}$ and $\mathbf{S}_2$ of $\mathbf{S}_{B_2}$, $\mathbf{S}_1 \otimes \mathbf{S}_2$ is a subset of $\mathbf{S}_{B_1 \cup B_2}$ and is defined as:

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

Note that $\mathbf{S}_1 \otimes \mathbf{S}_2$ 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 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 with the vertices being $\mathbf{S}_B$. 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 $\mathsf{ac}(B)^-$ and let $\mathsf{bas}(A)$ be its (strong) basin of attraction. Then

Definition 9 (TS of non-elementary blocks). The transition system of B realised by $\mathsf{bas}(A)$ is defined as a tuple $\mathsf{TS}_B = (\mathbf{S}, \rightarrow)$ where the set of states $\mathbf{S}$ of TS_B is a subset of $\mathbf{S}_{\mathsf{ac}(B)}$ such that $\mathbf{s} \in \mathbf{S}$ if and only if $\mathbf{s}|_{\mathsf{ac}(B)^-} \in \mathsf{bas}(A)$ and for any two states $\mathbf{s}, \mathbf{s}' \in \mathbf{S}_{\mathsf{ac}(B)}$ there is a transition $\mathbf{s} \rightarrow \mathbf{s}'$ if and only if either $\mathsf{hd}(\mathbf{s}, \mathbf{s}') = 1$ and $\mathbf{s}'[i] = f_i(\mathbf{s})$ where $i = \arg(\mathsf{hd}(\mathbf{s}, \mathbf{s}'))$ or $\mathsf{hd}(\mathbf{s}, \mathbf{s}') = 0$ and there exists i such that $\mathbf{s}'[i] = f_i(\mathbf{s})$.

4.4 The main results

In this section, we state and prove 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. This section can be skipped on a quick-read without affecting the continuity of the article.

We first show that the attractors of (the TS of) BN are preserved across the TSs of the blocks. We prove this by proving a series of lemmas which will form the basis of the final inductive argument. Let us start with the case where our given Boolean network BN has two basic blocks B_1 and B_2. We shall later generalise the results to the case where BN has more than two basic blocks by inductive arguments.

Note that either one or both the blocks B_1 and B_2 are elementary. If only one of the blocks is elementary, we shall, without loss in generality, assume that it is B_1. Let $\mathsf{TS}, \mathsf{TS}_1$ and TS_2 be the transition systems of BN, B_1 and B_2 respectively where, if B_2 is non elementary, we shall assume that TS_2 is the transition system of B_2 realised by the basin of an attractor A_1 of TS_1 as defined in Definition 9.

The states of a transition system will be denoted by $\mathbf{s}$ or $\mathbf{t}$ with appropriate subscripts and/or superscripts. For any state $\mathbf{s} \in \mathsf{TS}$ (resp. $\mathbf{t} \in \mathsf{TS}$), we shall denote $\mathbf{s}|_{B_1}$ (resp. $\mathbf{t}|_{B_1}$) by $\mathbf{s}_1$ (resp. $\mathbf{t}_1$) and $\mathbf{s}|_{B_2}$ (resp. $\mathbf{t}|_{B_2}$) by $\mathbf{s}_2$ (resp. $\mathbf{t}_2$). Similarly, for a set of states $\mathbf{T}$ of TS, $\mathbf{T}_1$ and $\mathbf{T}_2$ will denote the set of projections of the states in $\mathbf{T}$ to B_1 and B_2 respectively.

Let $B_1^- = B_1 \setminus (B_1 \cap B_2)$ and $B_2^- = B_2 \setminus (B_1 \cap B_2)$. We shall denote any transition $\mathbf{s} \longrightarrow \mathbf{s}'$ in TS by $\mathbf{s} \xrightarrow{B} \mathbf{s}'$ if the variable whose value changes in the transition is in the set B.

Lemma 4. For an elementary block B_i of BN and for every $\mathbf{s}_i, \mathbf{s}_i'$ of TS_i, if there is a path from $\mathbf{s}_i$ to $\mathbf{s}_i'$ in TS_i, then there is a path from $\mathbf{s}$ to $\mathbf{s}'$ in TS such that $\mathbf{s}|_{B_i} = \mathbf{s}_i, \mathbf{s}'|_{B_i} = \mathbf{s}_i'$ and $\mathbf{s}|_{B_j^-} = \mathbf{s}'|_{B_j^-}$, $j \neq i$.

Proof. Let B_i be elementary and suppose $\mathbf{s}_i^0 \xrightarrow{B_1} \mathbf{s}_i^1 \xrightarrow{B_1} \ldots \xrightarrow{B_1} \mathbf{s}_i^m$, where $\mathbf{s}_i^0 = \mathbf{s}_i$ and $\mathbf{s}_i^m = \mathbf{s}_i'$, be a path from $\mathbf{s}_i$ to $\mathbf{s}_i'$ in TS_i. Let $\mathbf{s}|_{B_j^-} = \mathbf{s}'|_{B_j^-} = \mathbf{s}_j^-$. It is clear that $(\mathbf{s}_i^0 \otimes \mathbf{s}_j^-) \xrightarrow{B_1} (\mathbf{s}_i^1 \otimes \mathbf{s}_j^-) \xrightarrow{B_1} \ldots \xrightarrow{B_1} (\mathbf{s}_i^m \otimes \mathbf{s}_j^-)$ is a path from $\mathbf{s}$ to $\mathbf{s}'$ in TS where $\mathbf{s} = (\mathbf{s}_i^0 \otimes \mathbf{s}_j^-), \mathbf{s}' = (\mathbf{s}_i^m \otimes \mathbf{s}_j^-)$ and $\mathbf{s}$ and $\mathbf{s}'$ have the required properties. Indeed, since B_i is elementary and values of the nodes in B_j^- are not modified along the path. □

Lemma 5. For every $\mathbf{s}, \mathbf{s}'$ of TS if there is a path from $\mathbf{s}$ to $\mathbf{s}'$ in TS then there is a path from $\mathbf{s}_i$ to $\mathbf{s}_i'$ in TS_i for every elementary block B_i.

Proof. Suppose $\rho = \mathbf{s}^0 \rightarrow \mathbf{s}^1 \rightarrow \ldots \mathbf{s}^m$, where $\mathbf{s}^0 = \mathbf{s}$ and $\mathbf{s}^m = \mathbf{s}'$ be a path from $\mathbf{s}$ to $\mathbf{s}'$ in TS. Let B_i be an elementary

block of BN. We inductively construct a path ρ_i from $\mathbf{s}_i$ to $\mathbf{s}'_i$ in TS_i using ρ. $\rho_i^j,\ 0 \leq j < m$, will denote the prefix of ρ_i constructed in the jth step of the induction. Initially $\rho_i^0 = \mathbf{s}_i^0$. Suppose ρ_i^j has been already constructed and consider the next transition $\mathbf{s}^j \rightarrow \mathbf{s}^{j+1}$ in ρ. If this transition is labeled with B_i then we let $\rho_i^{j+1} = \rho_i^j \xrightarrow{B_i} \mathbf{s}_i^{j+1}$. Otherwise if this transition is labeled with $B_j^-, j \neq i$, then we let $\rho_i^{j+1} = \rho_i^j$. Since by induction hypothesis ρ_i^j is a path in TS_i and we add to this a transition from ρ only if a node of the elementary block B_i is modified in this transition, such a transition exists in TS_i. Hence, ρ_i^{j+1} is also a path in TS_i. Continuing in this manner, we shall have a path from $\mathbf{s}_i$ to $\mathbf{s}'_i$ in TS_i at the last step when $j+1 = m$. □

Lemma 6. Suppose B_1 and B_2 are both elementary blocks and $B_1 \cap B_2 = \emptyset$. Then for every $\mathbf{s}, \mathbf{s}' \in \mathsf{TS}$, there is a path from $\mathbf{s}$ to $\mathbf{s}'$ in TS if and only if there is a path from $\mathbf{s}_i$ to $\mathbf{s}'_i$ in every TS_i.

Proof. Follows directly from Lemma 4 and Lemma 5. □

Lemma 7. Let B_1 and B_2 be two elementary blocks of BN, $B_1 \cap B_2 = \emptyset$. Then we have that A is an attractor of of TS if and only if there are attractors A_1 and A_2 of TS_1 and TS_2 resp. such that $A = A_1 \otimes A_2$.

Proof. Follows directly from Lemma 6. □

Lemma 8. Let BN have two blocks B_1 and B_2 where B_2 is non-elementary, B_1 is elementary and is the parent of B_2. Then we have A is an attractor of TS if and only if A_1 is an attractor of TS_1 and A is also an attractor of TS_2 where TS_2 is realized by $\mathsf{bas}(A_1)$.

Proof. Suppose A is an attractor of TS and for contradiction suppose A_1 is not an attractor of TS_1. Then either there exist $\mathbf{s}, \mathbf{s}' \in A$ such that there is no path from $\mathbf{s}_1$ to $\mathbf{s}'_1$ in TS_1. But that is not possible by Lemma 5. Or there exist $\mathbf{s}_1 \in A_1$ and $\mathbf{s}'_1 \notin A_1$ such that there is a transition from $\mathbf{s}_1$ to $\mathbf{s}'_1$. But then by Lemma 4, there is a transition from $\mathbf{s} \in A$ to $\mathbf{s}' \notin A$ in TS where $\mathbf{s}|_{B_1} = \mathbf{s}_1$ and $\mathbf{s}'|_{B_2} = \mathbf{s}'_2$. This contradicts the assumption that A is an attractor of A. Next suppose A is not an attractor of TS_2. Then there is a transition in TS_2 from $\mathbf{s} \in A$ to $\mathbf{s}' \notin A$. But we have, by the construction of TS_2 (Definition 9), that this is also a transition in TS which again contradicts the assumption that A is an attractor of TS.

For the converse direction, suppose for contradiction that A is an attractor of TS_2 and A_1 is an attractor of TS_1 but A is not an attractor of TS. We must then have that there is a transition in TS from $\mathbf{s} \in A$ to $\mathbf{s}' \notin A$. If this transition is labelled with B_1 then we must have, by Lemma 5, that there is a transition in TS_1 from $\mathbf{s}_1$ to $\mathbf{s}'_1$. But since $\mathbf{s}'_1 \notin A_1$ this contradicts the assumption that A_1 is an attractor of TS_1. Next, suppose that this transition is labelled with B_2^-. We must then have that $\mathbf{s}_1 = \mathbf{s}'_1 \in A_1$. Hence, by the construction of TS_2 (Definition 9) it must be the case that $\mathbf{s}' \in \mathsf{TS}_2$ and this transition from $\mathbf{s}$ to $\mathbf{s}'$ is also present in TS_2. But this contradicts the assumption that A is an attractor of TS_2. □

Now suppose BN has k blocks that are topologically sorted as $\{B_1, B_2, \ldots, B_k\}$. Note that for every i such that $1 \leq i \leq k$, $(\bigcup_{j \leq i} B_j)$ is an elementary block of BN and we denote its transition system by $\overline{\mathsf{TS}}_i$.

Theorem 2 (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 $\mathsf{bas}(\otimes_{j \in I} A_j)$, 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 $\mathsf{ac}(B_{i+1})^-$. 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 B_{i+1} and A is an attractor of TS_k.

Proof. The proof is by induction on i. The base case is when $i = 2$ and BN has two blocks B_1 and B_2. If B_1 and B_2 are both elementary then the result follows from Lemma 7. If B_1 is elementary and is the parent of B_2 then the result follows from Lemma 8.

For the inductive case suppose the result holds for some i where $2 \leq i < k$. Now both $(\bigcup_{j \in I} B_j)$, where I is the set of indices of the basic blocks in $\mathsf{ac}(B_{i+1})^-$, and $(\bigcup_{j \leq i} B_j)$ are elementary. Now, if B_{i+1} is elementary then the result follows from Lemma 7. If B_{i+1} is non-elementary then $(\bigcup_{j \in I} B_j)$ is the parent of B_{i+1} and the result follows from Lemma 8. □

Next, we show that the basins of attractions of the attractors of (the TS of) BN are preserved across the TSs of the blocks as well. We again do this by proving a series of lemmas leading up to the final inductive argument. Let us come back to the case where BN has two blocks B_1 and B_2.

Lemma 9. Suppose $B_1 \cap B_2 = \emptyset$ and both B_1 and B_2 are elementary blocks of BN. Let A, A_1 and A_2 be attractors of $\mathsf{TS}, \mathsf{TS}_1$ and TS_2 respectively where $A = A_1 \otimes A_2$. Then $\mathsf{bas}_{\mathsf{TS}}(A) = \mathsf{bas}_{\mathsf{TS}_1}(A_1) \otimes \mathsf{bas}_{\mathsf{TS}_2}(A_2)$.

Proof. Follows easily from Lemma 6. □

Lemma 10. Let A, A_1 and A_2 be the attractors of $\mathsf{TS}, \mathsf{TS}_1$ and TS_2 respectively where B_1 and B_1 are elementary and non-elementary blocks respectively of BN with B_1 being the parent of B_2 and TS_2 being realized by $\mathsf{bas}_{\mathsf{TS}_1}(A_1)$ and $A = A_2$. Then $\mathsf{bas}_{\mathsf{TS}_1}(A_1) \otimes \mathsf{bas}_{\mathsf{TS}_2}(A_2) = \mathsf{bas}_{\mathsf{TS}_2}(A_2) = \mathsf{bas}_{\mathsf{TS}}(A)$.

Proof. Since TS_2 is realized by $\mathsf{bas}_{\mathsf{TS}_1}(A_1)$, by its construction (Definition 9) we have, for every state $\mathbf{s} \in \mathsf{TS}_2$, $\mathbf{s}_1 \in \mathsf{bas}_{\mathsf{TS}_1}(A_1)$. Hence $\mathsf{bas}_{\mathsf{TS}_1}(A_1) \otimes \mathsf{bas}_{\mathsf{TS}_2}(A_2) = \mathsf{bas}_{\mathsf{TS}}(A_2)$.

We next show that $\mathsf{bas}_{\mathsf{TS}_2}(A_2) = \mathsf{bas}_{\mathsf{TS}}(A)$. Suppose $\mathbf{s} \in \mathsf{bas}_{\mathsf{TS}_2}(A_2)$. To show that $\mathbf{s} \in \mathsf{bas}_{\mathsf{TS}}(A)$, it is enough to show that: (i) There is a path from $\mathbf{s}$ to some $\mathbf{s}^A \in A$ in TS and (ii) there is no path from $\mathbf{s}$ to $\mathbf{t} \in A'$ for some attractor $A' \neq A$ of TS.

(i) Since $\mathbf{s} \in \mathsf{bas}_{\mathsf{TS}_2}(A_2)$, and $A_2 = A$, there is a path ρ from $\mathbf{s}$ to $\mathbf{s}^A \in A$ in TS_2. It is easy to see from the construction of TS_2 (Definition 9) that ρ is also a path in TS from $\mathbf{s}$ to $\mathbf{s}^A$.

(ii) Suppose for contradiction that there is a path ρ' in TS from $\mathbf{s}$ to $\mathbf{t} \in A'$ for some attractor $A' \neq A$ of TS. Since $A' \neq A$ we must have that either (a) $A_1 \neq A'_1$ or (b) $A_1 = A'_1$ but $A_2 \neq A'_2$.

(a) In this case, by Lemma 5, there must be a path from $\mathbf{s}_1$ to $\mathbf{t}_1 \in A'_1$ which is a contradiction to the fact that $\mathbf{s}_1 \in \mathsf{bas}(A_1)$.

(b) We have by Theorem 2 that $A'_2 = A'$. Once again from the construction of TS_2 (Definition 9) it is easy to see that ρ' is also a path in TS_2 from $\mathbf{s}$ to $\mathbf{t} \in A'$. But this contradicts the fact that $\mathbf{s} \in \mathsf{bas}_{\mathsf{TS}_2}(A_2)$.

For the converse direction suppose that $\mathbf{s} \in \mathsf{bas}_{\mathsf{TS}}(A)$. To show that $\mathbf{s} \in \mathsf{bas}_{\mathsf{TS}_2}(A_2)$, it is enough to show that: (iii) There is a path from $\mathbf{s}$ to some $\mathbf{s}^{A_2} \in A_2$ and (iv) there is no path from $\mathbf{s}$ to $\mathbf{t} \in A'_2$ for some attractor $A'_2 \neq A_2$ of TS_2.

(iii) Since $\mathbf{s} \in \mathsf{bas}_{\mathsf{TS}}(A)$, there is a path ρ in TS from $\mathbf{s}$ to some $\mathbf{s}^A \in A$. By the fact that $A_2 = A$ and by the construction of TS_2 (Definition 9) it is clear that ρ is also a path in TS_2 from $\mathbf{s}$ to $\mathbf{s}^A \in A_2$.

(iv) Suppose for contradiction that there is a path ρ' in TS_2 from $\mathbf{s}$ to $\mathbf{t} \in A'_2$ for some attractor $A'_2 \neq A_2$ of TS_2. By Theorem 2, A'_2 is equal to an attractor A' of TS and $A' \neq A$. It is then easy to see again from the construction of TS_2 (Definition 9) that ρ' is also a path in TS from $\mathbf{s}$ to $\mathbf{t} \in A'$. But this contradicts the assumption that $\mathbf{s} \in \mathsf{bas}_{\mathsf{TS}}(A)$. □

Let us, for the final time, come back to the case where BN has $k > 2$ blocks and these blocks are topologically sorted as $\{B_1, B_2, \ldots, B_k\}$. Let i range over $\{1, 2, \ldots, k\}$. By the theorem on attractor preservation, Theorem 2, we have that $(\otimes_{j\leq i} A_j)$ is an attractor of $\overline{\mathsf{TS}}_i$.

Lemma 11. Suppose BN has k basic blocks that are topologically sorted as $\{B_1, B_2, \ldots, B_k\}$. 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 $\mathsf{bas}(\otimes_{j\in I} A_j)$, its basin w.r.t. the TS for $(\bigcup_{j\in I} B_j)$, where I is the set of indices of the basic blocks in $\mathsf{ac}(B_{i+1})^-$ [where $(\otimes_{j\in I} A_j)$, by Theorem 2, is an attractor of the TS for $(\bigcup_{j\in I} B_j)$]. Then for every i, $(\otimes_{j\leq i} \mathsf{bas}_{\mathsf{TS}_j}(A_j)) = \mathsf{bas}(\otimes_{j\leq i} A_i)$ where $\mathsf{bas}(\otimes_{j\leq i} A_j)$ is the basin of attraction of $(\otimes_{j\leq i} A_j)$ with respect to transition system $\overline{\mathsf{TS}}_i$ of $(\bigcup_{j\leq i} B_j)$.

Proof. The proof is by induction on i. The base case is when $i = 2$. Then either B_1 and B_2 are both elementary and disjoint in which case the proof follows from Lemma 9. Or, B_1 is elementary and B_2 is non-elementary and B_1 is the parent block of B_2. In this case the proof follows from Lemma 10.

For the inductive case, suppose that the conclusion of the theorem holds for some $i : 2 \leq i < k$. Now, consider $(\otimes_{j\leq (i+1)} \mathsf{bas}_{\mathsf{TS}_j}(A_j))$. By the induction hypothesis, we have that $(\otimes_{j\leq i} \mathsf{bas}_{\mathsf{TS}_j}(A_j)) = \mathsf{bas}(\otimes_{j\leq i} A_j)$ where $(\otimes_{j\leq i} A_j)$ is an attractor of the transition system $\overline{\mathsf{TS}}_i$ of the elementary block $(\bigcup_{j\leq i} B_j)$ and $\mathsf{bas}(\otimes_{j\leq i} A_j)$ is its basin. Now, either B_{i+1} is elementary in which case we use Lemma 9 or B_{i+1} is non-elementary and $(\bigcup_{j\in I} B_j)$ is its parent in which case we use Lemma 10.

In either case, we have $(\otimes_{j\leq (i+1)} \mathsf{bas}_{\mathsf{TS}_j}(A_j)) = \mathsf{bas}(\otimes_{j\leq (i+1)} A_j)$, where $\mathsf{bas}(\otimes_{j\leq (i+1)} A_j)$ is the basin of attraction of the attractor $(\otimes_{j\leq (i+1)} A_j)$ of $\overline{\mathsf{TS}}_{i+1}$. □

Theorem 3 (Preservation of basins). Given the hypothesis and the notations of Theorem 2, we have $(\otimes_{i\leq k} \mathsf{bas}_{\mathsf{TS}_i}(A_i)) = \mathsf{bas}_{\mathsf{TS}}(A)$ where $\mathsf{bas}_{\mathsf{TS}}(A)$ is the basin of attraction of the attractor $A = (A_1 \otimes A_2 \otimes \ldots \otimes A_k)$ of TS.

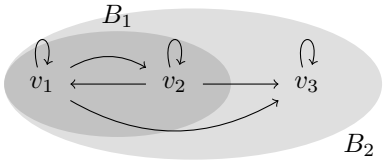
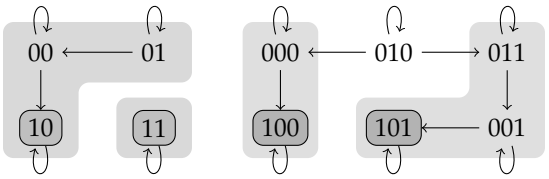


Fig. 2: The blocks of BN.



(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.

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

Proof. Follows directly by setting $i = k$ in Lemma 11. □

Example 4. Continuing with Example 1 and 2, 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 Fig. 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 Fig. 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 Fig. 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 2 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.

4.5 The decomposition-based algorithm

Equipped with the results in Theorems 2 and 3, we can describe our procedure for computing the strong basin of the target attractor based on decomposing the BN into smaller blocks. We shall then use this procedure to give an algorithm for the minimal control problem. Theorem 3 tells us that in order to compute $\mathsf{bas}(A_t)$ it is sufficient to compute the local basins of the projection of A_t to each block B_i (which by Theorem 2 is an attractor of B_i) and finally merge these local basins using the cross operation.

Now, as per the results in Section 4.4 for every non-elementary block B_i, its TS, TS_i, is realised by the basin of attraction $\mathsf{bas}(\otimes_{j\in I} A_j)$ of its ancestor blocks, having the index set I. Now since, by the structure of the block graph $\mathcal{G}_{\mathsf{BN}}$, for every j, $j < i$ and $j \notin I$, B_j is not an ancestor of B_i, we can, without loss of generality, assume that TS_i is realised by the basin of attraction $\mathsf{bas}(\otimes_{j<i} A_j)$ of all the blocks that precede B_i in the topological ordering of $\mathcal{B}$. The local basin of attraction of A_i is computed w.r.t. the transitions in TS_i, which, as we just saw, is in turn realised by $\mathsf{bas}(\otimes_{j<i} A_j)$. Hence we slightly modify the original

Algorithm 1b Relativised computation of strong basin

1: **procedure** COMP_SB_REL($\mathbf{f}$, A, Z)
2: WB := COMP_WB_REL($\mathbf{f}, A, Z$);
3: Initialise SB := $\emptyset$;
4: **while** SB $\neq$ WB **do**
5: If SB$\neq \emptyset$ do WB := SB;
6: SB:=WB$\setminus$(pre((post(WB) $\cap$ Z) $\setminus$ WB)$\cap$WB);
7: **end while**
8: **return** SB;
9: **end procedure**

procedure COMP_SB, for computing the strong basin of an attractor, by making it compute the basin *relative to* a given subset of states Z (say). This procedure is called COMP_SB_REL and is given in Algorithm 1b. It now takes as input not only the functions of the BN and an attractor A but also a subset of states Z. It calls a fixpoint procedure for the computation of the weak basin of A which is also relativised to the set Z, which is called COMP_WB_REL. The difference of the procedure in Algorithm 1b to that in Algorithm 1 is that states that are added to the final SB are also in the input set Z. This check is carried out in line 6.

Algorithm 3 finally implements the decomposition-based idea in pseudo-code. It takes as input the graph $\mathcal{G}_{\mathsf{BN}}$ and the update functions $\mathbf{f}$ of a given Boolean network, and an attractor A and returns the strong basin of attraction of A. Line 2 decomposes $\mathcal{G}_{\mathsf{BN}}$ into the blocks $\mathcal{B}$ (resulting in k blocks) using the procedure FORM_BLOCK from [21] and line 3 topologically sorts the blocks by constructing the block graph $\mathcal{G}_{\mathcal{B}}$. Lines 5-14 then cycle through the blocks of $\mathcal{B}$ in topological order and for each block B_i, line 6 decomposes the attractor A into its projection to B_i, denoted as A_i. If B_i is elementary then it computes the local strong basin SB_i of A_i independently, using the procedure COMP_SB of Algorithm 1 (line 8). If B_i is non-elementary, it computes its local strong basin SB_i relative to the basin of $(A_1 \otimes A_2 \otimes \ldots \otimes A_{i-1})$ using the procedure COMP_SB_REL of Algorithm 1b (line 11). Thus by Theorem 2, at every iteration i of the for-loop, the invariant that A_i is an attractor of TS_i is maintained. Line 13 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 3). It then easily follows that

***Proposition* 2.** Algorithm 3 correctly computes the strong basin of the attractor A.

We now plug the procedure COMP_SB_DECOMP of Algorithm 3 into Algorithm 2 to derive our decomposition-based minimal target control algorithm, Algorithm 4, from source state $\mathbf{s}$ to target attractor A_t.

Once again, it is worthwhile to note that our decomposition-based algorithm, Algorithm 4, can still be exponential in the size of the input BN, in the worst case. Indeed, since the weak basin of an attractor A of BN might be the entire TS. In such a case Algorithms 3 and 1b combine to end up computing the entire weak basin of A. This is no surprise as the problem in general is PSPACE-hard. However, in practice, such cases are rare, esp. for biological networks which, in addition, have highly modular structures, and our algorithm does provide considerable gains

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

1: **procedure** COMP_SB_DECOMP($\mathcal{G}_{\mathsf{BN}}$,$\mathbf{f}$,$A$)
2: $\mathcal{B}$:= FORM_BLOCK($\mathcal{G}_{\mathsf{BN}}$);
3: $\mathcal{B}$:= TOP_SORT($\mathcal{B}$);
4: k := size of $\mathcal{B}$;
5: **for** $i := 1$ to k **do**
6: A_i :=DECOMPOSE(A, B_i); *//Decompose the target attractor into block* B_i
7: **if** B_i is an elementary block **then**
8: SB_i :=COMP_SB($\mathbf{f}|_{B_i}, A_i$);
9: **else**
10: Z :=CROSS ($\mathsf{SB}, \{0,1\}^{|B_i|}$);
11: SB_i :=COMP_SB_REL($\mathbf{f}|_{\overline{B}_i}, (\otimes_{j \leq i} A_j), Z$);
12: **end if**
13: SB :=CROSS ($\mathsf{SB}, \mathsf{SB}_i$);
14: **end for**
15: **return** SB;
16: **end procedure**

Algorithm 4 Decomposition-based minimal simultaneous target control

1: **procedure** DECOMP_MINIMAL_CONTROL($\mathcal{G}_{\mathsf{BN}}, \mathbf{f}, \mathbf{s}, A_t$)
2: SB := COMP_SB_DECOMP($\mathcal{G}_{\mathsf{BN}}, \mathbf{f}, A_t$);
3: **return** $\arg(\mathsf{hd}(\mathbf{s},\mathsf{SB}))$;
4: **end procedure**

for such networks. This will be shown later in Section 5 in the case studies.

4.6 An optimisation of the algorithm

Looking back at our decomposition-based algorithm for the computation of the strong basin of a given attractor A, Algorithm 3, note that it does not take into consideration any information about the source state $\mathbf{s}$ of our STC problem. However it turns out that we can indeed use $\mathbf{s}$ to optimise our decomposition-based algorithm further. This optimisation proceeds as follows. For every block B_i, we check if the projection of $\mathbf{s}$ to B_i is in fact equal to the projection of our target attractor A to B_i. If that is the case, intuitively, we do not need to perturb the indices of $\mathbf{s}$ in B_i in the final control C that we compute for the STC problem.

The procedure OPT_COMP_SB_DECOMP of Algorithm 5 implements this idea in pseudo-code where it now takes as input also the source state $\mathbf{s}$ of the STC problem. The only difference with the procedure COMP_SB_DECOMP of Algorithm 3 is that inside the for-loop of lines 5-14, for every block B_i, the algorithm first computes the projection of the source state s in block B_i, denoted $\mathbf{s}_i$, at line 2. Then, it checks whether $\{\mathbf{s}_i\} = A_i$, if so, there is no need to compute the entire local basin of A_i for B_i. It then simply sets SB_i as the cross of SB and A_i, where SB is the (subset of the) basin computed thus far. If $\{\mathbf{s}_i\}$ is not equal to A_i, it computes the basin of A_i as in Algorithm 3 (lines 7-12). To save space, in the description of the procedure OPT_COMP_SB_DECOMP in Algorithm 5, we have only shown the additional lines that need to be added to the original COMP_SB_DECOMP procedure of Algorithm 3. Finally, in the algorithm for computing the STC, the procedure OPT_COMP_SB_DECOMP is used instead of COMP_SB_DECOMP in Algorithm 4.

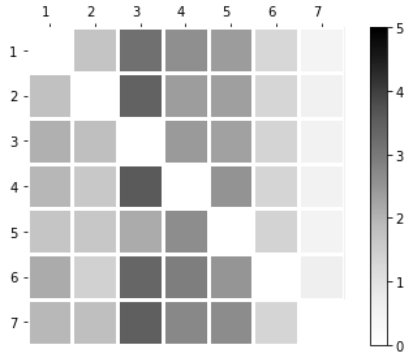
Algorithm 5 Optimised computation of strong basin

1: **procedure** OPT_COMP_SB_DECOMP($\mathcal{G}_{\text{BN}}$,$\mathbf{f}$,s,A)
2: $\{\mathbf{s}_i\} :=$ DECOMPOSE($\{\mathbf{s}\}, B_i$);
3: **if** $\{\mathbf{s}_i\} = A_i$ **then**
4: $\text{SB}_i :=$CROSS(SB, A_i);
5: **end if**
6: **end procedure**

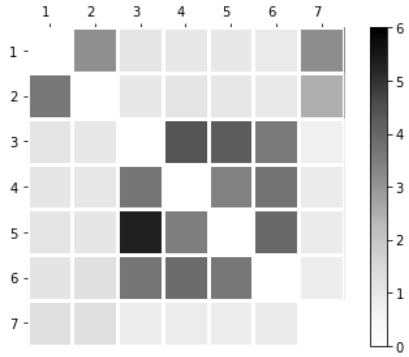
5 CASE STUDIES

To demonstrate the correctness and efficiency of our control framework, we evaluate the performance of the global approach, the decomposition-based approach, and the optimised decomposition-based approach on several networks. The approaches described by Algorithm 2, Algorithm 4, and Algorithm 5 are implemented in the software tool ASSA-PBN [22], which is based on the model checker [23] to encode BNs into the efficient data structure, binary decision diagrams (BDDs). All the experiments are performed on a computer (MacBook Pro), which contains a CPU of Intel Core i7 @3.1 GHz and 8 GB of DDR3 RAM.

We apply three approaches on six real-life biological networks [24], [25], [26], [27], [28], [29] and three randomly generated networks (BN-100, 120, 180). For every pair of source and target attractors of the networks, we compute a minimal control C that can realise the minimal simultaneous single-step target control as explained in Section 3.4. An overview of the networks and their evaluation results is given in Table 1[3].



(a) $speedups = T_g/T_d$



(b) $speedups = T_d/T_o$

Fig. 4: Speedups of the PC12 cell network.

3. The symbol $*$ denotes that the algorithm fails to return any results within one hour.

Exact minimal control set. According to Proposition 1 and 2, our decomposition-based approach and the global approach compute an exact minimal control C with respect to a source state and a target attractor. The numbers of driver nodes computed by these approaches are identical, demonstrating the correctness of our decomposition-based approaches. Moreover, the number of driver nodes is relatively small compared to the size of the network.

Table 2 summaries the Hamming distances (HD) between attractors and the number of driver nodes (#D) for all pairs of source and target attractors for the PC12 cell network. The attractors are labelled with numbers. The numbers in the first column and the first row represent the source and target attractors, respectively. 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. In particular, to drive the network from any other attractor to the attractor 'cell differentiation', only one node 'NGF' is required, which is consistent with the conclusion in [25].

Efficiency and scalability. In Table 1, T_g, T_d and T_o represent the total time costs of computing a minimal control for all pairs of source and target attractors with the global approach, the decomposition-based approach and the optimised decomposition-based approach, respectively. The key of the control approaches lies in computation of the basin of the target attractor. Benefited from the fixpoint computation of strong basin, as described in Algorithm 1, all the approaches are efficient. Compared with the global approach, our decomposition-based approach has an advantage in terms of efficiency (see the speedups in column T_g/T_d) for *structurally well-behaved* networks, thanks to its 'divide and conquer' strategy. And the efficiency of the decomposition-based approach is further improved by the optimisation (see the speedups in column T_d/T_o), since the optimisation compares the source and the target attractors to avoid unnecessary computation of the local strong basins of the target attractor in some blocks.

Fig. 4a and Fig. 4b show the heatmaps of speedups gained by the decomposition-based approach over the global approach (T_g/T_d) and speedups gained by the optimisation (T_d/T_o) for every pair of source and target attractors of the PC12 cell network. The indices of the columns and rows are consistent with Table 2. In Fig. 4a, the speedups gained by the decomposition-based approach is highly relevant to the target attractor, as most of the computation time is spent on the computation of its strong basin. The speedups gained by the optimisation is determined by both the source and the target attractors. If the Hamming distance between the source and the target attractor is small, the optimisation can skip local strong basin computation in many blocks and gain a high speedup. For instance, the dark gray entities in Fig. 4b indicate high speedups and the Hamming distances between their associated source and target attractors (see Table 2) are relatively small.

Finally, our decomposition-based approaches have better scalability than the global approach, the global approach fails to compute the results for large networks (BN-120 and BN-180 with 120 and 180 nodes, resp.) as it deals with the entire networks at once.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2019.2915081, IEEE/ACM Transactions on Computational Biology and Bioinformatics

networks	# nodes	# blocks	# attractors	time (seconds)			speedups	
				T_g	T_d	T_o	T_g/T_d	T_d/T_o
tumour	32	13	9	2.34	1.87	1.48	1.25	1.26
PC12	32	19	7	0.45	0.26	0.20	1.70	1.34
hematopoiesis	33	11	5	1.99	1.25	0.77	1.60	1.61
HGF	66	26	18	236.64	132.33	66.90	1.79	1.98
bortezomib	67	28	5	208.92	35.97	13.16	5.81	2.73
T-diff	68	34	12	48.16	21.21	10.70	2.27	1.98
BN-100	100	36	9	2491.47	22.24	5.47	112.05	4.07
BN-120	120	27	4	$*$	61.83	31.18	$*$	1.98
BN-180	180	62	2	$*$	25.94	1.40	$*$	18.52

TABLE 1: An overview of the networks and their evaluation results.

attractors	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 2: The Hamming distance between attractors and the number of driver nodes of the PC12 cell network.

6 Conclusion and Future Work

In this work, we have described a decomposition-based approach towards the computation of a minimal set of nodes of a Boolean network that needs to be simultaneously controlled in order 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 a 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. In this section, we conclude by looking back critically at our approaches, 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, one might ask in retrospect, what is the inherent characteristic of our decomposition-based approaches 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 are 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 in the network cannot be too few. Otherwise, our decomposition-based approaches 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. On the other hand, if a network has too many blocks, computing local strong basins for all the blocks may hamper the efficiency of the decomposition-based approach. For such kind of networks, our optimisation of the decomposition-based approach, proposed in Algorithm 5 provides a solution by avoiding the expensive computation of the local strong basins of the blocks in which the source state and the target attractor do not differ. Another way to reduce the number of 'small' blocks 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. In [30], we have proposed a method for the near-optimal decomposition of BNs and we are working further on it to find an optimal 'block-to-node ratio', given which, our decomposition-based control approaches fare the best. We believe there are many real-life networks which might benefit from the optimal decomposition techniques before the computation of the control.

As mentioned in the section on related work, the control approaches based on computation of the feedback vertex set [10], [11], [15] and the stable motifs [12] are promising approximate control algorithms for nonlinear dynamical networks. In the near future, we would like to compare our approaches with these two in terms of efficiency and the number of driver nodes. We also plan to investigate if and how the concept of simultaneous single-step control that we study here can be cast into the framework of Boolean control networks (BCNs) [17], [18], [19], [20]. We can then analyse if we can compute control strategies by appropriately modyfing the techniques already available in the literature for BCNs. Conversely, the decomposition-based approach that we developed here might in turn be useful in improving some of the control techniques on BCNs esp. on large networks. Finally, we would like to extend our decomposition-based approach to the control of probabilistic Boolean networks [31], [32].

Acknowledgments

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

REFERENCES

[1] T. Graf and T. Enver, "Forcing cells to change lineages," *Nature*, vol. 462, no. 7273, pp. 587–594, 2009.

[2] A. d. Sol and N. Buckley, "Concise review: A population shift view of cellular reprogramming," *Stem Cells*, vol. 32, no. 6, pp. 1367–1372, 2014.

[3] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Controllability of complex networks," *Nature*, vol. 473, pp. 167–73, 2011.

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

[5] Y.-C. Lai, "Controlling complex, non-linear dynamical networks," *National Science Review*, vol. 1, no. 3, pp. 339–341, 2014.

[6] S. A. Kauffman, "Homeostasis and differentiation in random genetic control networks," *Nature*, vol. 224, pp. 177–178, 1969.

[7] S. Huang, "Genomics, complexity and drug discovery: insights from Boolean network models of cellular regulation," *Pharmacogenomics*, vol. 2, no. 3, pp. 203–222, 2001.

[8] H. Mandon, S. Haar, and L. Paulevé, "Relationship between the reprogramming determinants of boolean networks and their interaction graph," in *Proc. 5th Workshop on Hybrid Systems Biology*, ser. LNCS, vol. 9957. Springer, 2016, pp. 113–127.

[9] ——, "Temporal reprogramming of Boolean networks," in *Proc. 15th Conference on Computational Methods in Systems Biology*, ser. LNCS, vol. 10545. Springer, 2017, pp. 179–195.

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

[11] B. Fiedler, A. Mochizuki, G. Kurosawa, and D. Saito, "Dynamics and control at feedback vertex sets. i: Informative and determining nodes in regulatory networks," *Journal of Dynamics and Differential Equations*, vol. 25, no. 3, pp. 563–604, 2013.

[12] J. G. T. Zañudo and R. Albert, "Cell fate reprogramming by control of intracellular network dynamics," *PLoS Computational Biology*, vol. 11, no. 4, p. e1004193, 2015.

[13] E. Czeizler, C. Gratie, W. K. Chiu, K. Kanhaiya, and I. Petre, "Target controllability of linear networks," in *Proc. 14th Conference on Computational Methods in Systems Biology*, ser. LNCS, vol. 9859. Springer, 2016, pp. 67–81.

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

[15] J. G. T. Zañudo, G. Yang, and R. Albert, "Structure-based control of complex networks with nonlinear dynamics," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 114, no. 28, pp. 7234–7239, 2017.

[16] A. Gates and L. Rocha, "Control of complex networks requires both structure and dynamics," *Scientific Reports*, vol. 6, 2016.

[17] H. Qi and D. Cheng, "Analysis and control of boolean networks: a semi-tensor product approach," in *Asian Control Conference, 2009. ASCC 2009. 7th*. IEEE, 2009, pp. 1352–1356.

[18] J. Zhong, D. W. Ho, J. Lu, and W. Xu, "Global robust stability and stabilization of boolean network with disturbances," *Automatica*, vol. 84, pp. 142–148, 2017.

[19] D. Laschov, M. Margaliot, and G. Even, "Observability of boolean networks: A graph-theoretic approach," *Automatica*, vol. 49, no. 8, pp. 2351–2362, 2013.

[20] N. Bof, E. Fornasini, and M. E. Valcher, "Output feedback stabilization of boolean control networks," *Automatica*, vol. 57, pp. 21–28, 2015.

[21] A. Mizera, J. Pang, H. Qu, and Q. Yuan, "Taming asynchrony for attractor detection in large Boolean networks," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (Special issue of APBC'18)*, 2018.

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

[23] A. Lomuscio, H. Qu, and F. Raimondi, "MCMAS: An open-source model checker for the verification of multi-agent systems," *International Journal on Software Tools for Technology Transfer*, vol. 19, no. 1, pp. 9–30, 2017.

[24] D. P. A. Cohen, L. Martignetti, S. Robine, E. Barillot, A. Zinovyev, and L. Calzone, "Mathematical modelling of molecular pathways enabling tumour cell invasion and migration," *PLoS Computational Biology*, vol. 11, no. 11, p. e1004571, 2015.

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

[26] S. Collombet, C. van Oevelen, W. Ortega, J.L.S.and Abou-Jaoudé, B. Di Stefano, M. Thomas-Chollier, T. Graf, and D. Thieffry, "Logical modeling of lymphoid and myeloid cell specification and transdifferentiation," *Proceedings of the National Academy of Sciences*, vol. 114, no. 23, pp. 5792–5799, 2017.

[27] A. Singh, J. M. Nascimento, S. Kowar, H. Busch, and M. Boerries, "Boolean approach to signalling pathway modelling in HGF-induced keratinocyte migration," *Bioinformatics*, vol. 28, no. 18, pp. 495–501, 2012.

[28] V. Chudasama, M. Ovacik, D. Abernethy, and D. Mager, "Logic-based and cellular pharmacodynamic modeling of bortezomib responses in u266 human myeloma cells," *Journal of Pharmacology and Experimental Therapeutics*, vol. 354, no. 3, pp. 448–458, 2015.

[29] A. Naldi, J. Carneiro, C. Chaouiya, and D. Thieffry, "Diversity and plasticity of th cell types predicted from regulatory network modelling," *PLoS Computational Biology*, vol. 6, no. 9, 2010.

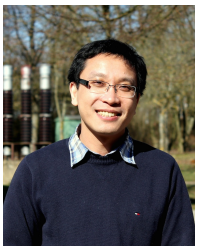
[30] C. Su, J. Pang, and S. Paul, "Towards optimal decomposition of Boolean networks," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (Special issue of APBC'19)*, 2019.

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

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

Dr. Soumya Paul received his PhD in Theoretical Computer Science from the Institute of Mathematical Sciences, Chennai, India in 2012. Currently, he is a Research Associate at the Faculty of Science, Technology and Communications at the University of Luxembourg. His research interests include logic and strategic reasoning and computational systems biology.

Cui Su received her MSc degrees in Systems Engineering from Yanshan University in 2016. She is currently a PhD student at the Interdisciplinary Centre for Security, Reliability and Trust at the University of Luxembourg, working on the research project SEC-PBN. Her research interests lie in network control and computational systems biology.



Dr. Jun Pang received his PhD in Computer Science from Vrije Universiteit Amsterdam, The Netherlands in 2004. Currently, he is a senior researcher in the Computer Science and Communications research unit at the University of Luxembourg. His research interests include formal methods, security and privacy, social media mining, and computational systems biology.

Dr. Andrzej Mizera received the MSc degree in Computer Science from the University of Warsaw, Poland in 2005. He then obtained his PhD in Computer Science with minor in mathematics in the area of Computational Systems Biology from the Åbo Akademi University, Turku, Finland in 2011. He is currently a research associate at the Institute of Computer Science, Polish Academy of Sciences, Poland. His research interests are related to computational and mathematical modelling of biological systems.