Bioinformatics, 35, 2019, i558–i567
doi: 10.1093/bioinformatics/btz371
ISMB/ECCB 2019

Controlling large Boolean networks with single-step perturbations

Alexis Baudin[1], Soumya Paul[2], Cui Su[3] and Jun Pang[2,3,*]

[1]Department of Computer Science, École Normale Supérieure Paris-Saclay, 94230 Cachan, France, [2]Faculty of Science, Technology and Communication, University of Luxembourg, 4365 Esch-sur-Alzette, Luxembourg and [3]Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, 1855 Luxembourg, Luxembourg

*To whom correspondence should be addressed.

Abstract

Motivation: The control of Boolean networks has traditionally focussed on strategies where the perturbations are applied to the nodes of the network for an extended period of time. In this work, we study if and how a Boolean network can be controlled by perturbing a minimal set of nodes for a single-step and letting the system evolve afterwards according to its original dynamics. More precisely, given a Boolean network (BN), we compute a minimal subset C_{min} of the nodes such that BN can be driven from any initial state in an attractor to another 'desired' attractor by perturbing some or all of the nodes of C_{min} for a single-step. Such kind of control is attractive for biological systems because they are less time consuming than the traditional strategies for control while also being financially more viable. However, due to the phenomenon of state-space explosion, computing such a minimal subset is computationally inefficient and an approach that deals with the entire network in one-go, does not scale well for large networks.

Results: We develop a 'divide-and-conquer' approach by decomposing the network into smaller partitions, computing the minimal control on the projection of the attractors to these partitions and then composing the results to obtain C_{min} for the whole network. We implement our method and test it on various real-life biological networks to demonstrate its applicability and efficiency.

Contact: jun.pang@uni.lu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

In control theory, a dynamical system is controllable if, through an appropriate manipulation of a few parameters, it can be driven from any initial state to any desired final state within finite time. Although control theory is a mathematically highly developed branch of engineering with applications to electric circuits, manufacturing processes, communication systems, robots etc., fundamental questions pertaining to the controllability of complex biological networks have resisted rapid advances. The reasons for this are 3-fold. First, biological networks tend to be large with an exponential increase in combinatorial complexity with the addition of every parameter or interaction which in turn effects their controllability. This is often referred to as the 'dimensionality problem' (Hecker *et al.*, 2009). Secondly, such networks are highly non-linear with switch-like interactions between the components. It is unclear how the linear functions usually studied in traditional control theory could capture such dynamics (Tyson *et al.*, 2001, 2003; Zañudo and Albert, 2015). And finally, the notion of controllability in biological systems is different from the classical definition of linear controllability. In such systems, rather than controlling single states, the control of collective dynamic behaviour may be more feasible (Wang *et al.*, 2016).

The recent discoveries in cell reprogramming have rekindled the interest in the control of cellular behaviour and biological systems in general. Cell reprogramming is a way to change one cell phenotype to another, allowing tissue or neuron regeneration techniques. Current studies have shown that differentiated adult cells can be reprogrammed to an embryonic-like pluripotent state or directly to other types of adult cells without the need of intermediate reversion to a pluripotent state (Graf and Enver, 2009; Sol and Buckley, 2014). 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. Such medicines target specific proteins within the cellular systems aiming to drive it from any state to a desired phenotype. This motivates the question of identifying multiple drug targets using which the network can be 'controlled', i.e. driven from any state to any desired target. Furthermore, for the feasibility of the synthesis of such drugs, the number of such targets should be minimized. However, as already mentioned, biological

networks are intrinsically large (number of components, parameters, interactions, etc.) which results in an exponentially increasing number of potential drug target combination making a purely experimental approach quickly infeasible. This reinforces the need for mathematical modelling and efficient computational techniques.

Boolean networks (BNs), first introduced by Kauffman (1969), are a popular and well-established framework for modelling gene regulatory networks and their associated signalling pathways. Its main advantage is that it is simple and is yet able to capture the important dynamical properties of the system under study (D'haeseleer *et al.*, 2000), thus facilitating the modelling of large biological systems as a whole. 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 hypothesized to characterize cellular phenotypes (Kauffman, 1969) and also correspond to functional cellular states such as proliferation, apoptosis, differentiation, etc. (Huang, 2001). The *control* of a BN therefore refers to the reprogramming/changing of the parameters of the BN (functions, values of variables, etc.) so that its dynamics eventually reaches a desired attractor or steady state.

The control of linear networks is a well-studied problem (Kalman, 1963) and such control strategies have been proposed over the years. Recent work on network controllability has shown that the control and reprogramming of intercellular networks can be achieved by a small number of control targets (Kim *et al.*, 2013). The control of such networks can have two objectives: to drive the dynamics to (i) a single desired target attractor of the network irrespective of the current state. We shall call such a control *target control* or TC, (ii) any attractor of the network irrespective of the current state. We shall call this type of control *full control* or FC.

Now, biological networks (both intracellular and intercellular) are intrinsically non-linear and the strategies developed for the control of linear networks do not directly apply to these networks. Moreover, networks with non-linear dynamics are arguably more complex with many feed-forward and feedback loops for both activation and inhibition. This might explain why there has not been a lot of work on the control problem for non-linear networks. For the target control problem, Kim *et al.* (2013) developed a method to identify the so-called 'control kernel', which is a minimal set of nodes for driving a synchronous BN into a desired attractor. Their method is based on the construction of the full state transition graph of the network and as such does not scale well for large networks. Zhao *et al.* (2016) developed a network graph aggregation approach to control synchronous BNs. These two methods, however, are not applicable for asynchronous BNs. For the control of asynchronous BNs, Zañudo and Albert (2015) developed an extended period control method to identify a set of nodes based on the 'stable motifs' (SM) of the network to drive the network towards a desired target attractor. For the problem of full control, Fiedler *et al.* (2013; Mochizuki *et al.*, 2013; Zañudo *et al.*, 2017) developed a method for controlling networks, whose dynamics are governed by ordinary differential equations (ODEs) by computing the feedback vertex set (FVS) of the corresponding dependency graph. It is however unclear how their method can be lifted to the discrete switch-like dynamics of BNs.

The control strategies in the above and most of the methods studied in the literature have one thing in common—the perturbation is applied continuously for an extended period of time. However, there are and can be obvious drawbacks to such a strategy. For example, the concentration of the complexes (drugs, viruses, etc.) applied for the perturbations might fall below the requisite threshold over time in which case it needs to be administered again and again to maintain appropriate levels. For example, *half-life* or *decay rates* exist for almost any substance that is ever added to cells—whether it is a drug or a nutrient or a virus—and there will be degradation due to temperature, evaporation, depletion by the cells. etc. This is discussed, for example in Michels and Frei (2013) where they mention the decay of ascorbate in cell culture medium. For the case of adding virus to cells, the depletion of active virus in the cell culture dish happens relatively fast and can therefore be a limiting factor for inserting a potential gene (say) into the target cells via the virus. This typical issue of low *transduction efficiency* is often counteracted by adding the virus to the cell repeatedly (e.g. see Zhu *et al.*, 2015). Such repeated administration of the virus is also called for when the experimenter wants to target multiple cells instead of just one (Charrier *et al.*, 2011; Hofherr *et al.*, 2017). The phenomenon also occurs when inserting smaller copies of gene into the cell without integrating it into the genome, which does not require the help of a virus. Even in such cases, the experimenter has to try to add the gene-copies repeatedly since the genes are not attached to the cell's genome (Cervera *et al.*, 2015). The repeated addition of the complexes to the cell thus requires constant monitoring of the system over an extended period of time. Furthermore, the complexes themselves are difficult and expensive to acquire prohibiting their extensive use.

Thus a more short-term control strategy might be well suited for biological networks (Cornelius *et al.*, 2013). In this work we explore such a control strategy where the perturbation is applied for a single time-step (read instantaneously) and the system is left to evolve on its own, according to its original dynamics. For both versions of the control problem, TC and FC, we develop a method to identify an exact minimal set $\mathsf{C}_{\min}$ of nodes of a given Boolean network BN, such that the above controls can be achieved by perturbing some of the nodes in $\mathsf{C}_{\min}$. Such short-term control strategies have been studied in the literature (Cornelius *et al.*, 2013), where a control method based on simulations for large networks has been proposed. Although the ideas presented in Cornelius *et al.* (2013) are quite relevant to those we use here, their methods do not directly compare to the ones that we develop in this work. Indeed, since first, they deal with ODE networks, and not Boolean networks. And secondly, since there does not yet exist ways to compute the basins of attractions of ODE networks, their method is based on simulations where the search is automatically terminated if the system is not controlled within a sufficiently large number of iterations. On the other hand, we can indeed compute efficiently the attractors and basins of BNs using methods developed in-house, and hence can compute the 'exact' minimal control for a given BN.

It is well-known that the precise identification of control parameters and control strategies of non-linear networks must exploit *both* their structural and dynamic properties (Gates and Rocha, 2016). This rules out purely structure-based methods for identifying the exact control subset, like that of Liu *et al.* (2011), which has been shown to either overshoot or undershoot the control subset for different networks (Gates and Rocha, 2016). The dynamics of a Boolean network BN is given in terms of its *transition system*, which as we already observed is exponential in the size of BN itself. Any non-simulation-based algorithm that purely exploits this dynamics by working on entire BN in one-go has to, in principle, work with the full transition system, and thus has limited scalability. As the BN grows in size, the number of possible behaviours (traces) grows exponentially with it (state-space explosion). This means that even any simulation-based algorithm has to deal with a very large number of

traces to preserve their guaranteed accuracy. This, in turn, limits their efficiency as well.

Our algorithm takes the approach of 'divide-and-conquer' whereby it decomposes the network into smaller partitions, computes the minimal set of control nodes in each of these partitions and then composes the results to obtain the set $\mathsf{C}_{\min}$ for the whole network. While doing the composition, the algorithm crucially needs to check whether there exist subsets of $\mathsf{C}_{\min}$ that can be perturbed in the starting state that results in a state that belongs to the 'basin of attraction' of the target attractor(s), from which there only exist paths towards the target attractor and there is no path leading to any other attractor of the network. We therefore assume that the algorithm is able to call the efficient procedure developed in Paul *et al.* (2018) to compute the basin of attraction of an attractor of BN. It is worth noting that our algorithm always computes an *exact* minimal set of control nodes.

We have implemented our algorithm and tested it on a variety of real-life biological networks modelled as BNs. We also compared our results with the existing approaches for the control of non-linear networks. For TC, we compared our method with the stable motifs based control (SM) of Zañudo and Albert (2015) and for FC, we evaluate its performance without any comparison as, to the best of our knowledge, no method for the full control of asynchronous BNs exists in the literature. Our findings can be summarized as follows: For TC, our method outperforms the SM based method in terms of efficiency (for almost all the networks). For FC, our method can compute the minimal full control set efficiently. The advantage of our method is that we give the exact strategy to be applied for the control given any source state and any target attractor. We particularly note that, even for very large networks, the subset of control nodes identified for both control strategies forms a relatively small set which is a desirable property for the control of such networks.

2 Background and notations

Let $N = \{1, 2, \ldots, n\}$ where $n \geq 1$. 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 is a Boolean variable and $\mathbf{f} = (f_1, f_2, \ldots, f_n)$ is a tuple of Boolean functions over $\mathbf{x}$. In what follows, i will always range over N, unless stated otherwise. A Boolean network $\mathsf{BN} = (\mathbf{x}, \mathbf{f})$ may be viewed as a directed graph $\mathcal{G}_{\mathsf{BN}} = (V, E)$, called the *dependency graph* of BN, where $V = \{v_1, v_2 \ldots, v_n\}$ is the set of *vertices* or *nodes* (intuitively, v_i corresponds to the variable x_i for all i) and for every $i, j \in N$, there is a directed edge from v_j to v_i, often denoted as $v_j \to v_i$, if and only if f_i depends on x_j. Thus V is ordered according to the ordering of $\mathbf{x}$. The *structure* of BN refers to the structure of its dependency graph. For any vertex $v_i \in V$, we let $\mathrm{ind}(v_i) = i$ be the index of v_i in this ordering. For any subset W of V, $\mathrm{ind}(W) = \{\mathrm{ind}(v) | \ v \in W\}$. For the rest of the exposition, we assume an arbitrary but fixed network BN of n variables is given to us and $\mathcal{G}_{\mathsf{BN}} = (V, E)$ is its associated dependency 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 $\mathrm{hd}(\mathbf{s}, \mathbf{s}')$ and $\arg(\mathrm{hd}(\mathbf{s}, \mathbf{s}')) \subseteq N$ 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 $\mathrm{hd}(\mathbf{s}, \mathbf{S}') = \min_{\mathbf{s}' \in \mathbf{S}'} \mathrm{hd}(\mathbf{s}, \mathbf{s}')$. We let $\arg(\mathrm{hd}(\mathbf{s}, \mathbf{S}'))$ denote the set of subsets of N such that $I \in \arg(\mathrm{hd}(\mathbf{s}, \mathbf{S}'))$ if and only if I is a set of indices of the variables that realize $\mathrm{hd}(\mathbf{s}, \mathbf{S}')$.

The behaviour of BN is captured by its *evolution dynamics* which is defined as follows. Initially, BN is in a state $\mathbf{s}_0$ and its state changes in every discrete time-step according to the update functions $\mathbf{f}$. In this work, we shall be exclusively concerned with the asynchronous updating scheme but all our results transfer to the synchronous updating scheme as well. Suppose $\mathbf{s}_0 \in \mathbf{S}$ is an initial state of BN. The *asynchronous evolution* of BN is a function $\xi : \mathbb{N} \to \wp(\mathbf{S})$ 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* of $\mathbf{s}$, 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 $\mathbf{s}'[i] = f_i(\mathbf{s})$. Note that the asynchronous dynamics is non-deterministic.

The dynamics of a Boolean network can be represented as a *state transition graph* or a *transition system (TS)*. The *transition system* of BN, denoted by the generic notation TS is a tuple $(\mathbf{S}, \to)$ 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 $\mathbf{s} \to \mathbf{s}'$, if and only if $\mathbf{s}'$ is a possible next state of $\mathbf{s}$. 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 a state $\mathbf{s} \in \mathbf{S}$, $\mathrm{reach}_{\mathsf{TS}}(\mathbf{s})$ denotes the set of states $\mathbf{s}'$ such that there is a path from $\mathbf{s}$ to $\mathbf{s}'$ in TS. An *attractor* A of TS (or of BN) is a minimal subset of states of $\mathbf{S}$ such that for every $\mathbf{s} \in A$, $\mathrm{reach}_{\mathsf{TS}}(\mathbf{s}) = A$. A 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. For an attractor A of TS, the *basin* of attraction of A, denoted $\mathrm{bas}_{\mathsf{TS}}(A)$, is a subset of states of $\mathbf{S}$ such that $\mathbf{s} \in \mathrm{bas}_{\mathsf{TS}}(A)$ if $\mathrm{reach}_{\mathsf{TS}}(s) \cap A \neq \emptyset$ and $\mathrm{reach}_{\mathrm{TS}}(\mathrm{s}) \cap A' = \emptyset$ for any attractor $A' \neq A$ of BN. A *control* C is a (possibly empty) subset of 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. Henceforth, we shall drop the subscripts TS when no ambiguity arises.

Control problems. Let BN be a given Boolean network, $\mathbf{S}$ be the set of states of BN and $\mathcal{A}$ be the set of all its attractors. We are interested in the following kinds of control on BN. Note that for us, the control is applied in a single time-step (hence simultaneously) to the current state $\mathbf{s}$ under consideration and the system is let to evolve as per its original dynamics afterwards.

1. **Source-target control** (STC)**:** Let $\mathbf{s} \in \mathbf{S}$ and let $A_t \in \mathcal{A}$ be a target attractor, A control $\mathsf{C}^{\mathbf{s} \to A_t}$ is an STC for $\mathbf{s}$ and A_t if, after the application of $\mathsf{C}^{\mathbf{s} \to A_t}$ to $\mathbf{s}$, BN eventually reaches A_t.
2. **Target control** (TC)**:** Let $A_t \in \mathcal{A}$ be a target attractor. A control $\mathsf{C}^{\to A_t}$ is a TC for A_t if for any attractor $A_s \in \mathcal{A}, A_s \neq A_t$, and for any state $\mathbf{s} \in A_s$, there exists a subset $\mathsf{C}_{\mathbf{s}}$ of $\mathsf{C}^{\to A_t}$ such that $\mathsf{C}_{\mathbf{s}}$ is an STC of $\mathbf{s}$ for A_t.
3. **Full control** (FC)**:** A control C is an FC for BN if for any pair of attractors $A_s, A_t \in \mathcal{A}, A_s \neq A_t$, and for any state $\mathbf{s} \in A_s$, there exists a subset $\mathsf{C}^{\mathbf{s} \to A_t}$ of C such that $\mathsf{C}^{\mathbf{s} \to A_t}$ is an STC of $\mathbf{s}$ for A_t.

Given the above kinds of control, we are interested in the following control problems on a non-linear, asynchronous BN:

1. **min-STC problem:** Given BN, a source state $\mathbf{s}$ and a target attractor $A_t \in \mathcal{A}$, find a minimal STC. Such an STC will be called a min-STC and denoted as $\mathsf{C}_{\min}^{\mathbf{s} \to A_t}$.
2. **min-TC problem:** Given BN, and a target attractor $A_t \in \mathcal{A}$, find a minimal TC. Such a TC will be called a min-TC and denoted as $\mathsf{C}_{\min}^{\to A_t}$.

3. **min-FC problem:** Given BN and the set of attractors $\mathcal{A}$, find a minimal FC for BN. Such a control will be called a min-FC and denoted as $\mathsf{C}_{\min}$.

In Paul *et al.* (2018), we developed a decomposition-based approach for the efficient solution to the min-STC problem [item (1) above] for large BNs exploiting both their structure and dynamics. We showed that the efficient computation of the minimal control given a target attractor A_t boils down to the efficient computation of the basin, $\text{bas}(A_t)$ of A_t. We therefore developed an algorithm for the computation of $\text{bas}(A_t)$ by decomposing the BN into connected components called *blocks*, computing the local basins of the projections of A_t to each of these blocks and then eventually merging these local basins to obtain $\text{bas}(A_t)$. We demonstrated both efficiency and effectiveness of our approach on different real-life biological networks. In this work, we shall target the control problems (2) and (3) listed above. Note that for control problem (3), we assume that the set of attractors $\mathcal{A}$ of BN is already given to us. If however, $\mathcal{A}$ is not known, we first need to compute $\mathcal{A}$ from BN for which we have already developed and implemented efficient procedures (see e.g. Mizera *et al.*, 2017, 2018; Yuan *et al.*, 2016). In the algorithms that we develop here, we shall use the procedure to compute the basin of a given attractor A of a given Boolean network developed in Paul *et al.* (2018) and shall refer to it as Compute_Basin(A).

3 Results

Towards the solution of control problems 2 and 3 above, we first define a generic control problem which we call the **Minimal All-Pairs Control**.

- **Minimal All-Pairs Control (min-APC):** Let $\mathcal{A}$ be the set of all attractors of BN and let $\mathcal{A}_s, \mathcal{A}_t \subseteq \mathcal{A}$ be subsets of attractors, called *source* and *target* attractors respectively. A control $\mathsf{C}^{\mathcal{A}_s \to \mathcal{A}_t}$ is an APC for $\mathcal{A}_s$ and $\mathcal{A}_t$ if for any pair of attractors $A_s \in \mathcal{A}_s, A_t \in \mathcal{A}_t, A_s \neq A_t$ and any state $\mathbf{s} \in A_s$, there exists $\mathsf{C}^{\mathbf{s} \to A_t} \subseteq \mathsf{C}^{\mathcal{A}_s \to \mathcal{A}_t}$ such that $\mathsf{C}^{\mathbf{s} \to A_t}$ is an STC of $\mathbf{s}$ for A_t. An APC which is minimal is called a min-APC and is denoted as $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$. The min-APC problem is then: given BN, $\mathcal{A}_s$ and $\mathcal{A}_t$, find a min-APC.

The control problems min-TC and min-FC are special cases of the min-APC problem when $\mathcal{A}_t$ is a singleton and when $\mathcal{A}_s = \mathcal{A}_t = \mathcal{A}$, respectively.

We first observe that the min-APC problem is computationally at least as hard as the min-STC problem. Indeed, since the min-STC problem for a source state $\mathbf{s}$ and a target attractor A_t, where $\mathbf{s}$ is a fixpoint attractor, is a special case of the min-APC problem where $\mathcal{A}_s = \{\{\mathbf{s}\}\}$ and $\mathcal{A}_t = \{A_t\}$. Since min-STC is already hard for PSPACE (Mandon *et al.*, 2016; Paul *et al.*, 2018), efficient solutions for min-APC are highly unlikely.

To gain an intuition into the problem, suppose all the attractors in $\mathcal{A}_s$ are singleton states (fixed points). Suppose, $A_s = \{\mathbf{s}\} \in \mathcal{A}_s$ is a source attractor and $A_t \in \mathcal{A}_t$ is a target attractor. It can be easily observed that the BN eventually and surely reaches A_t following the update dynamics, after a control C is applied to $\mathbf{s}$, if and only if $\mathsf{C}(\mathbf{s}) \in \text{bas}(A_t)$ (Paul *et al.*, 2018). Also, for any state $\mathbf{t} \in \text{bas}(A_t)$, the number of nodes to perturb to move from $\mathbf{s}$ to $\mathbf{t}$ is $\text{hd}(\mathbf{s}, \mathbf{t})$ and these nodes are given as $\arg(\text{hd}(\mathbf{s}, \mathbf{t}))$. So, let M be a $|\mathcal{A}_s| \times |\mathcal{A}_t|$ matrix such that for every pair of attractors $A_s \in \mathcal{A}_s$ and $A_t \in \mathcal{A}_t$, the (A_s, A_t)th entry of M, $\mathsf{M}[A_s, A_t]$ is a set of subsets of N such that for any subset $Z \subseteq N$, $Z \in \mathsf{M}[A_s, A_t]$ if and only if there exists $\mathbf{t} \in \text{bas}(A_t)$ such that $Z = \arg(\text{hd}(\mathbf{s}, \mathbf{t}))$. $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$ is then a minimal subset of N such that there exists a subset of $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$ in $\mathsf{M}[A_s, A_t]$ for every pair of attractors $A_s \in \mathcal{A}_s$ and $A_t \in \mathcal{A}_t$.

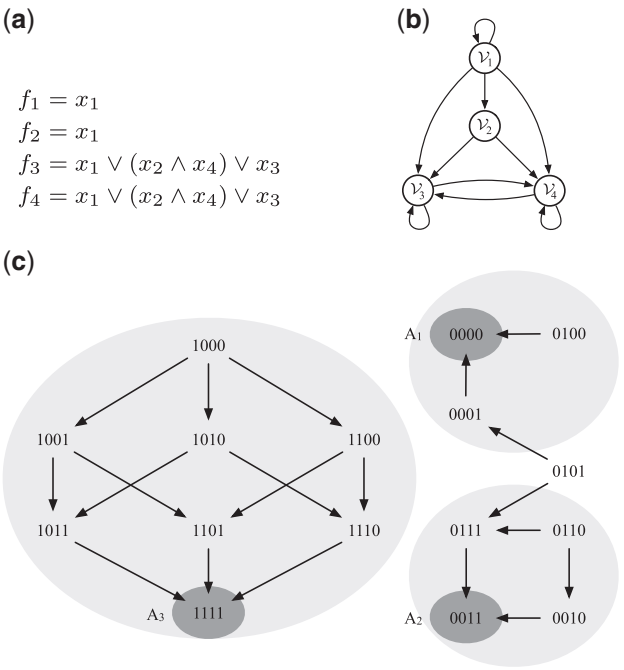

Fig. 1. (**a**) Boolean functions, (**b**) dependency graph and (**c**) TS for Example 1. The basins of attractions of the respective attractors are shown as shaded grey regions

Table 1. The matrix showing the indices to be controlled for pairs of attractors

	A_1	A_2	A_3
A_1	$\varnothing$	{{3}, {2, 3}, {3, 4}, {2, 3, 4}}	{{1}, {1, 2}, {1, 3}, {1, 4}, {1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {1, 2, 3, 4}}
A_2	{{3}, {3, 4}, {2, 3, 4}}	$\varnothing$	{{1}, {1, 2}, {1, 3}, {1, 4}, {1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {1, 2, 3, 4}}
A_3	{{1, 2, 3}, {1, 3, 4}, {1, 2, 3, 4}}	{{1}, {1, 2}, {1, 4}, {1, 2, 4}}	$\varnothing$

Example 1. *Consider a Boolean network* $\mathsf{BN} = (\mathbf{x}, \mathbf{f})$ *where* $\mathbf{x} = (x_1, x_2, x_3, x_4)$ *and* $\mathbf{f} = (f_1, f_2, f_3, f_4)$ *where* $f_1 = f_2 = x_1$ *and* $f_3 = f_4 = x_1 \vee (x_2 \wedge x_4) \vee x_3$. *The dependency graph of* BN *and its TS is shown in Figure 1. We suppress the self loops present in each of the states of the TS to avoid clutter. It has 3 single-state attractors* $\mathcal{A} = \{A_1, A_2, A_3\}$ *shown as dark grey nodes, where* $A_1 = \{0000\}, A_2 = \{0011\}, A_3 = \{1111\}$. *The basins of attractions of the respective attractors are shown as shaded grey regions.*

Table 1 shows the matrix M *that notes the indices of the variables that need to be changed to move from an attractor* A_s *in* $\mathcal{A}$ *to the basin of another attractor* A_t *in* $\mathcal{A}$. *From* M *we see that both the sets* {1, 2, 3} *and* {1, 3, 4} *are* min-FCs. *However,* {2, 3, 4}, *for example, is not a* min-FC *since it is not possible to move to the basin of* A_1 *from* A_3 *by perturbing only* v_2, v_3 *and* v_4.

Algorithm 1. All-pairs control

```
1:  procedure All_Pairs_Control (BN = (x, f), A_s, A_t, m)
2:      i := 0, APC := {}, k := ⌈n/m⌉, partitions := {V_1, V_2, ..., V_k}
3:      for j in [1, k] do                                   //initialize CMIN_0^j for all j
4:          CMIN_0^j := Min_Control(BN, A_s, A_t, V_j)       //min full control for V_j
5:          size_min(j) := |C^j| where C^j ∈ CMIN_0^j        //minimum size of the control set on V_j
6:      end for
7:      while APC = {} do
8:          for a_1, ..., a_k ≥ 0 such that a_1 + ··· + a_k = i do
9:              possAPC := {C_1 ∪ ... ∪ C_k | C_i ∈ CMIN_{a_i}^(i), i ∈ {1, ..., k}}   //for all possible controls of combined size i
10:             for C ∈ possAPC do
11:                 if Is_Control(C, BN, A_s, A_t) then APC := APC ∪ {C}   //check if it is a valid APC for BN
12:                 end if
13:             end for
14:         end for
15:         if APC = {} then                                 //if a valid APC for BN has not yet been found
16:             i ← i + 1;                                   //increase the size of the potential APC by 1
17:             for j = 1 to k do
18:                 CMIN_i^j :=Fixed_Control(BN, A_s, A_t, V_j, size_min(j) + i)   //look for an APC of the new size
19:             end for
20:         end if
21:     end while
22:     return APC
23: end procedure
```

We propose an algorithm based on the approach of 'divide-and-conquer' wherein we decompose the network into smaller partitions and solve the min-APC problem on these partitions. We then combine the results to obtain the control set for the entire network. We show that using such an approach, we can solve the problem on large Boolean networks arising from real-life biological systems much more efficiently compared with a global approach that works on the entire network in a single go. Towards that, we first need the notion of *projection* of a state to a subset of nodes of BN.

Let $V' = \{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\}$ be a subset of V, the projection of $\mathbf{s}$ to V', denoted $\mathbf{s}|_{V'}$ is an element of $\{0,1\}^k$ defined as $\mathbf{s}|_{V'} = (s[i_1], s[i_2], \ldots, s[i_k])$. The projection operation is lifted to a subset $\mathbf{S}'$ of $\mathbf{S}$ as $\mathbf{S}'|_{V'} = \{s|_{V'} \mid \mathbf{s} \in \mathbf{S}'\}$. A *decomposition* of BN is defined as a partitioning $V_1, V_2, \ldots V_k$ of V. Each $V_j, 1 \leq j \leq k$ will be called a *partition* of BN. For any attractor $A \in \mathcal{A}$ and for any partition V_j, $A|_{V_j}$ and $\text{bas}(A)|_{V_j}$ are well-defined. Given sets of source and target attractors $\mathcal{A}_s$ and $\mathcal{A}_t$, respectively, and a partition V_j, $\mathsf{C}^j \subseteq \text{ind}(V_j)$ is an APC on V_j if it satisfies the all-pairs control properties on BN projected to V_j. That is, for all $A_s \in \mathcal{A}_s$ and $A_t \in \mathcal{A}_t$, $A_s \neq A_t$ implies for all $\mathbf{s} \in A_s$, there exists $\mathsf{C} \subseteq \mathsf{C}^j$ such that $\mathsf{C}(\mathbf{s}|_{V_j}) \in \text{bas}(A_t)|_{V_j}$. The idea of the algorithm is based on the following proposition.

Proposition 1. *Let $\mathcal{A}_s$ and $\mathcal{A}_t$ be sets of source and targets attractors of* BN *and let $V_1, V_2, \ldots, V_k$ be a decomposition of V. If $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$ is a* min-APC *of* BN *then $(\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} \cap \text{ind}(V_j))$ is a* min-APC *on partition Vj for all $1 \leq j \leq k$. Furthermore, $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} = \cup_{1 \leq j \leq k}(\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} \cap \text{ind}(V_j))$.*

Proof. Suppose that $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$ is an APC of BN. Then by definition, for every pair of attractors $A_s \in \mathcal{A}_s$ and $A_t \in \mathcal{A}_t$, and for all $\mathbf{s} \in A_s$, there exists $\mathsf{C}_\mathbf{s} \subseteq \mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$ such that $\mathsf{C}_\mathbf{s}(\mathbf{s}) \in \text{bas}(A_t)$. This implies, for every partition V_j, $(\mathsf{C}_\mathbf{s} \cap \text{ind}(V_j))(\mathbf{s}|_{V_j}) \in \text{bas}(A_t)|_{V_j}$. Now, it must hold that $(\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} \cap \text{ind}(V_j)) = \cup_{A_s \in \mathcal{A}_s, A_t \in \mathcal{A}_t} \cup_{\mathbf{s} \in A_s}(\mathsf{C}_\mathbf{s} \cap \text{ind}(V_j))$. Thus, by definition, $(\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} \cap \text{ind}(V_j))$ is an APC on V_j. Moreover, since the partitions are mutually disjoint, we have $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} = \cup_{1 \leq j \leq k}(\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} \cap \text{ind}(V_j))$.

Next, suppose $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$ is also a minimal APC of BN but there exist some V_ℓ such that $(\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} \cap \text{ind}(V_\ell))$ is not a minimal APC on V_ℓ. Let C_ℓ be a minimal APC on V_ℓ such that $|\mathsf{C}_\ell| < |\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} \cap \text{ind}(V_\ell)|$. Then, from above, we have that there is another control $\hat{\mathsf{C}}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} = (\cup_{1 \leq j \leq k, j \neq \ell}(\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t} \cap \text{ind}(\mathsf{V}_j)) \cup \mathsf{C}_\ell)$ which is a minimal APC of BN and $|\hat{\mathsf{C}}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}| < |\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}|$, since the partitions are mutually disjoint. But this contradicts the minimality of $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$.

3.1 Main algorithm

We now describe our Algorithm 1, to solve the min-APC problem. The algorithm takes as input the functions of a Boolean network BN, sets of source and target attractors $\mathcal{A}_s$ and $\mathcal{A}_t$ and the size m of partitions that BN will be decomposed into and works as follows. It first computes and stores the basins of attractors of the attractors in $\mathcal{A}_t$ using the procedure Compute_Basin developed in Paul *et al.* (2018). It randomly decomposes BN into $k = \lceil |V|/m \rceil$ partitions $V_1, V_2, \ldots V_k$ each of size at most m (line 2 of Algorithm 1). For each partition V_j it computes the set of min-APCs, CMIN_0^j, on V_j using the helper function Min_Control (line 4). Let $r = \sum_{j=1}^k |\mathsf{C}^j|$ where $\mathsf{C}^j \in \mathsf{CMIN}_0^j$. By Proposition 1, we know that the size of a min-APC, $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$ for BN is at least r. The algorithm chooses one min-APC from each partition and checks if their union is a valid APC on the entire network BN by using the helper function Is_Control which queries the basins of attractions of the target attractors already computed. This is done in lines 8–14. If it cannot find an APC of size r, it increases the value of r by 1 and repeats the process: for each partition V_j it computes the set of APCs of the next larger size on V_j using the helper function Fixed_Control (line 18). It checks if there is a union of APC from each of the partitions the sizes of which sum to the new value of r and such that it forms a valid APC on BN. It repeats this process each time increasing the value of r by 1 till it finds a min-APC for BN, $\mathsf{C}_{\min}^{\mathcal{A}_s \to \mathcal{A}_t}$ (lines 15–20). The correctness of the algorithm is therefore trivially guaranteed.

Algorithm 2. Helper functions

```
1:  procedure Min_Control (BN, A_s, A_t, V_j)
2:      CMIN^j := {}, max = |V_j|, success := TRUE
3:      for i = 0 to max do
4:          for C ⊆ ind(V_j), |C| = i do                    //for all subsets C of size at most max of the indices in V_i
5:              for A_s ∈ A_s, A_t ∈ A_t, A_s ≠ A_t do
6:                  for s ∈ A_s do
7:                      if ¬(∃C' ⊆ C, C'(s|_{V_j}) ∈ bas(A_t)|_{V_j}) then success:= FALSE   //check if there exists a subset of C such that applying it to
8:                      end if                              //the projection of s to V_j results in a state in the projection
9:                  end for                                 //of bas(A_t) to V_j
10:             end for
11:             if success = TRUE then
12:                 CMIN^j := CMIN^j ∪ {C},  max := |C|     //if a control has been found, max is set to its size
13:             end if
14:         end for
15:     end for
16:     return CMIN^j
17: end procedure
18: procedure Fixed_Control(BN, A_s, A_t, V_j, m)
19:     CMIN^j := {}, success := TRUE
20:     for C ⊆ ind(V_j), |C| = m do                        //the potential control is of a fixed size m
21:         for A_s ∈ A_s, A_t ∈ A_t, A_s ≠ A_t do
22:             for s ∈ A_s do
23:                 if ¬(∃C' ⊆ C, C'(s|_{V_j}) ∈ bas(A_t)|_{V_j}) then success := FALSE
24:                 end if
25:             end for
26:         end for
27:         if success = TRUE then CMIN^j := CMIN^j ∪ {C}   //a valid control on V_j has been found
28:         end if
29:     end for
30:     return CMIN^j
31: end procedure
32: procedure Is_Control (C, BN, A_s, A_t)
33:     success := TRUE
34:     for A_s ∈ A_s, A_t ∈ A_t, A_s ≠ A_t do
35:         for s ∈ A_s do
36:             if ¬(∃C' ⊆ C, C'(s) ∈ bas(A_t)) then success := FALSE   //check if there is a subset of C which is a valid APC on BN
37:             end if
38:         end for
39:     end for
40:     return success
41: end procedure
```

We next describe the procedures Min_Control, Is_Control and Fixed_Control (Algorithm 2) used in Algorithm 1. We assume that the basins for all the attractors in $\mathcal{A}_t$ has been computed using the procedure Compute_Basin developed in Paul *et al.* (2018) and stored in an appropriate global data structure and can be accessed by all these procedures. For $A_t \in \mathcal{A}_t$, bas(A_t) will denote the basin of A_t as computed using Compute_Basin.

Min_Control takes as input the description of the Boolean network, the sets of the source and the target attractors and a partition V_j and it returns the min-APCs on partition V_j. To do that it first computes the projection to V_j of every state $\mathbf{s} \in A_s$ for every $A_s \in \mathcal{A}_s$ and of every $A_t \in \mathcal{A}_t$. Then for i from 0 to $|V_j|$, it checks if any subset C of ind(V_j) of size i satisfies the APC properties on V_j. That is, if for all $A_s \in \mathcal{A}_s$ and $A_t \in \mathcal{A}_t$, $A_s \neq A_t$ implies for all $\mathbf{s} \in A_s$, $\mathsf{C}(\mathbf{s}|_{V_j}) \in \text{bas}(A_t)|_{V_j}$. It returns all such subsets of size i (for the lowest value of i) and exits.

The procedure Fixed_Control is similar to Min_Control except that it returns an APC on V_j of size size_min(j) + i if it exists. Otherwise, it returns the empty set.

Is_Control checks if the given subset C is indeed an APC for $\mathcal{A}_s$ and $\mathcal{A}_t$. It does so by verifying if for all $A_s \in \mathcal{A}_s$ and all $\mathbf{s} \in A_s$ and for all $A_t \in \mathcal{A}_t$, $A_s \neq A_t$, there exists a subset $\mathsf{C_s}$ of C such that $\mathsf{C_s}(\mathbf{s}) \in \text{bas}(A_t)$.

As explained in Section 3, TC and FC are special cases of the APC problem. Thus, we compute $\mathsf{C}_{\min}^{\rightarrow \mathcal{A}_t}$ and $\mathsf{C}_{\min}$ with Algorithm 1 by setting $\mathcal{A}_t = A_t$ and $\mathcal{A}_s = \mathcal{A}_t = \mathcal{A}$, respectively.

We explain the working of Algorithm 1 here with a representative example.

Example 2. *Continuing with the Boolean network of Example 1, suppose now that we divide the vertices V of* BN *into two partitions,* $V_1 = \{v_1, v_2\}$ *and* $V_2 = \{v_3, v_4\}$. *The projections to these partitions of the attractors in* $\mathcal{A}$ *and their respective basins are given in Table 2.*

The algorithm works as follows. In Step 1, it computes the min-APC *sets for the projections to the partitions* V_1 *and* V_2 *as* $\mathsf{C}_1^1 = \{\{1\}\}$ *and* $\mathsf{C}_2^1 = \{\{3\}\}$, *respectively. Combining* C_1^1 *and* C_2^1 *we get* {1, 3} *but the check Is_Control returns that* {1, 3} *is not a valid full control for the whole network. So the algorithm moves to Step 2,*

Table 2. The projections of the attractors and basins to V_1 and V_2

$V_1 = \{v_1, v_2\}$		$V_2 = \{v_3, v_4\}$	
Attractor	Basin	Attractor	Basin
00	00, 01	00	00, 01
00	00, 01	11	11, 10
11	11, 10	11	11,10,01,00

where it looks for controls of size 3. For that it needs to find APCs *of size 2 in the projections to each of the partitions V_1 and V_2 and check the combinations of these and the controls C_1^1 and C_2^1 computed in Step 1, to find a control for the whole network. The* APCs *of size 2 that it finds for the two partitions in Step 2 are $\mathsf{C}_1^2 = \{\{1,2\}\}$ and $\mathsf{C}_2^2 = \{\{3,4\}\}$. Combining C_1^1 and C_2^2 we get* {1, 3, 4} *and combining C_2^1 and C_1^2 we get* {1, 2, 3} *both of which are valid* APCs *for* BN *which are also* FCs *in this example. Hence, the size of a minimum* FC *is 3.*

Remark. We make a quick remark on the computational complexity of our algorithm. Note that the algorithm can, in the worst case, take time exponential in the size of its input, which is the BN, the source and target attractors and the partition size. One way in which this can happen is, for example, when Is_Control in line 11 of Algorithm 1 returns FALSE for exponentially many potential controls before finding a valid APC. This, in turn, occurs when although each of the local controls $\mathsf{C}_1, \mathsf{C}_2, \ldots, \mathsf{C}_k$ are valid APCs on the partitions $V_1, V_2, \ldots, V_k$ but their union C is not a valid APC for the entire BN (the resulting state does not belong to the strong basin of some target attractor in $\mathcal{A}_t$). However, as we see in Section 4, such a case is extremely rare for BNs constructed for real-life biological networks. For such networks, Is_Control succeeds to find a valid APC within 2–3 iterations. This makes our procedure quite efficient on such networks.

4 Evaluation

As discussed in Section 1, the control method based on the computation of stable motifs (SM) (Zañudo and Albert, 2015) is a method of control applied for an extended period for the target control of asynchronous BNs. In this section, we compare our single-step control method for the min-TC problem (which we simply call TC) with SM even though the control computed by our method is applied only for a single time-step. Regarding the full control of asynchronous BNs, as we are not aware of any previous work in the literature that deals with the exactly same problem, we simply evaluate the performance of our method to compute the min-FC of a BN (which we simply call FC henceforth) to demonstrate its potential.

We apply these methods to 10 biological networks (Cohen *et al.*, 2015; Conroy *et al.*, 2014; Grieco *et al.*, 2013; Kim *et al.*, 2013; Naldi *et al.*, 2010; Offermann *et al.*, 2016; Remy *et al.*, 2015; Saez-Rodriguez *et al.*, 2007; Schlatter *et al.*, 2009; Singh *et al.*, 2012). Our methods for the computation of min-TC and min-FC are implemented as part of the software tool ASSA-PBN (Mizera *et al.*, 2018). All the experiments are performed on a computer with a CPU of Intel Core i7 @3.1 GHz and 8 GB of DDR3 RAM.

Description of the networks. We first describe the networks under study.

- The myeloid differentiation network is designed to model myeloid differentiation from common myeloid progenitors to megakaryocytes, erythrocytes, granulocytes and monocytes (Krumsiek *et al.*, 2011). This network has 11 nodes and 6 attractors, 4 of which agrees with microarray expression profiles of two different studies.
- The tumour network is built to study the role of individual mutations or their combinations in the metastatic process (Cohen *et al.*, 2015). This network contains 32 nodes and 9 attractors, which are consistent with Cohen *et al.* (2015).
- The PC12 cell network models the temporal sequence of protein signalling, transcriptional response and subsequent autocrine feedback (Offermann *et al.*, 2016). It has 33 nodes and 7 attractors.
- The bladder cancer network allows one to identify the deregulated pathways and their influence on bladder tumourigenesis (Wang *et al.*, 2012). It has 35 nodes. When the input nodes EGFR_stimulus and Growth_inhibitors are set to ON and DNA_damage is set to OFF, the network has four attractors: three correspond to growth arrest and one corresponds to cell proliferation.
- The MAPK network is constructed to study the MAPK responses to different stimuli and their contributions to cell fates (Grieco *et al.*, 2013). In this paper, we use the MPAK mutant r3, which has 53 nodes and 20 attractors.
- The model for HGF-induced keratinocyte migration captures the onset and maintenance of hepatocyte growth factor-induced migration of primary human keratinocytes (Singh *et al.*, 2012). It has 66 nodes and 18 attractors.
- The Th-cell differentiation network models the regulatory network and the signalling pathways controlling Th-cell differentiation (Naldi *et al.*, 2010). It consists of 68 nodes and 12 attractors with the same initial condition as mentioned in Naldi *et al.* (2010).
- The model of T-cell receptor signalling describes the complex signalling network governing the activation of T-cells via several receptors, including the T-cell receptor, the CD4/CD8 co-receptor, and the accessory signalling receptor CD28 (Saez-Rodriguez *et al.*, 2007). It has 95 nodes and 16 attractors are detected under certain conditions.
- The apoptosis network captures the central intrinsic and extrinsic apoptosis pathways and the pathways connected with them (Schlatter *et al.*, 2009). It has 97 nodes and 32 attractors when the nodes FASL_2, IL_1, TNF, UV, UV_2, FASL are fixed to OFF.
- The $CD4^+$ T-cell network allows us to study the downstream effects of $CAV1^{+/+}$, $CAV1^{+/-}$ and $CAV1^{-/-}$ on cell signalling and intracellular networks (Conroy *et al.*, 2014). This network is comprised of 188 nodes and 12 attractors under certain initial conditions.

An overview of the networks is given in Table 3. (We refer the sizes of the basins of attractors to the Supplementary Material.)

Selection of the partition size. We perform experiments on the biological networks described above to find out the best size of partitions for TC and FC. Since TC is a special case of FC, we only perform experiments for FC by setting the maximum size of a partition from 1 to 20 and comparing the time costs.

Figure 2 shows the normalized time costs with different sizes of partitions for the 10 networks. When the size equals 3, FC has the best efficiency for most of the networks. Hence, we set the partition size $m = 3$ except for the TC of HGF-induced keratinocyte migration, which is explained later.

Effectiveness. As illustrated in Proposition 1, our computation methods TC and FC identify the *minimal* control sets for single-step control. SM is an extended period control and it does not guarantee

Table 3. An overview of the networks and a comparison of the three methods on the control sets

Network	Nodes	Edges	Attractors	C^{TC}_{min}	C_{SM}	∩	C^{FC}_{min}
Myeloid	11	30	6	3	3	2	8
Tumour	32	158	9	2	*	*	14
PC12	33	62	7	1	1	1	15
Bladder	35	116	4	1	1	1	16
MAPK	53	105	20	4	4	4	20
HGF	66	103	18	4	*	*	34
Th-diff	68	175	12	3	2	2	17
T-cell	95	159	16	4	4	4	4
Apoptosis	97	192	32	5	5	5	5
CD4+	188	380	12	4	3	3	5

Note: ∩ represents the overlaps between C^{TC}_{min} and C_{SM}. The symbol '*' means the method fails to compute the results within 12 h.

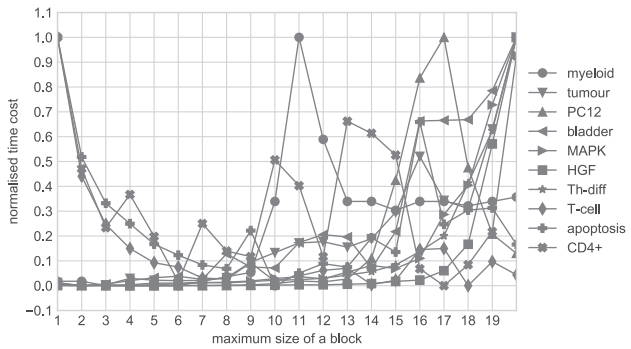

Fig. 2. Influence of the block size on the efficiency of FC

the minimality of the control sets as mentioned in Zañudo and Albert (2015).

Table 3 gives the sizes of the control sets computed by the three methods. It is worth noting that SM may capture unnecessary nodes. Taking the myeloid differentiation network as an example, Figure 3 gives the control nodes required by TC and SM to drive the network towards one of the attractors. The grey rectangular node—required by SM solely—has the same value in all the attractors, thus there is no need to control it.

Columns C^{TC}_{min} and C_{SM} are the number of driver nodes for one of the attractors computed by TC and SM. We can see that the results computed by the two methods are very close (see column ∩ in Table 3). Compared with SM, TC may lead to slightly larger control sets, like in the results of the Th-cell differentiation network and the $CD4^+$ T-cell network, due to the application of different control strategies—SM focuses on extended period control while we use single-step control. Despite that, the number of control nodes for single-step control are still small relative to the sizes of the networks.

The column C^{FC}_{min} describes the number of driver nodes required for the full control of the networks. For most of the networks, C^{FC}_{min} is much larger than C^{TC}_{min}. Three large networks (the T-cell network, the apoptosis network and the $CD4^+$ T-cell network) have small control sets because the attractors are caused by few nodes. For instance, the 32 attractors of the apoptosis network result from all combinations of values of five input nodes, i.e. 2^5. Even though it has 97 nodes and 32 attractors, by controlling the five input nodes, we can gain full control of the network.

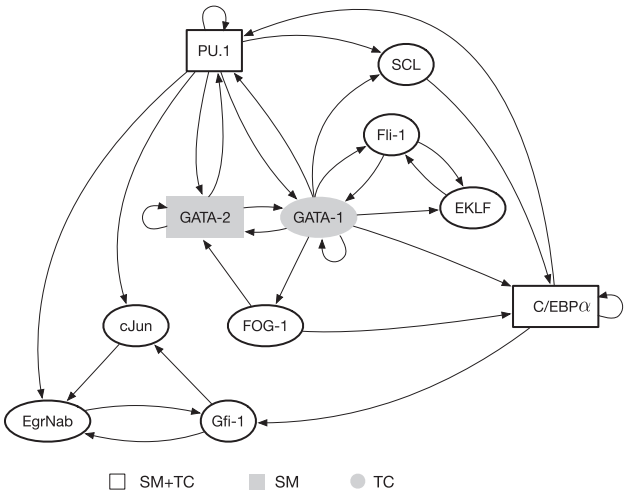

Fig. 3. The results of TC and SM on the myeloid differentiation network

Table 4. The time costs of the three control methods (TC, FC and SM)

Network	TC and FC				SM	
	T_{att}	T_{bas}	$T_{C^{TC}_{min}}$	$T_{C^{FC}_{min}}$	T_{att}	$T_{C_{SM}}$
Myeloid	0.002	0.004	0.004	0.001	6.989	7.846
Tumour	0.622	1.009	0.177	0.028	*	*
PC12	0.019	0.146	0.017	0.009	97.211	263.249
Bladder	0.881	0.318	0.813	0.745	26.955	32.587
MAPK	2.175	9.409	0.404	0.270	53.354	436.898
HGF	2.552	23.571	860.776	1.164	104.447	*
Th-diff	3.664	17.347	0.824	0.282	121.821	400.043
T-cell	2.170	14.762	0.565	0.335	58.418	9.967
Apoptosis	11.285	1230.200	1.778	1.045	222.241	55.578
CD4+	182.185	948.667	1.850	1.613	60.525	30.894

Note: Units of time are in seconds.

Efficiency. Table 4 gives the execution time of the three methods. Note that the partition size m only has influence on $T_{C^{TC}_{min}}$ and $T_{C^{FC}_{min}}$. The attractors and their basins are computed with methods in Mizera *et al.* (2019) and Paul *et al.* (2018) and their computation time may increase as the sizes of the networks increase.

$T_{C^{TC}_{min}}$ and $T_{C_{SM}}$ are the total time costs for computing the target control sets for all attractors of the networks. In general, our computation method TC outperforms SM in terms of efficiency for most of the networks. For the $CD4^+$ T-cell network, SM is faster than our method on attractor detection, mainly due to the fact that this network is sparse and has a simple structure. But this is rare for biological networks, as they are necessarily dense to performs remarkably robust regulatory tasks (Adai *et al.*, 2004; Blanchini and Franco, 2011).

The $T_{C^{TC}_{min}}$ of HGF-induced keratinocyte migration shows that the iteration of Algorithm 1 (lines 7–21) can be very time consuming. Taking one of the attractors as an example, the initial $r = \sum_{j=1}^{k} |C^j|$ is 13 and C^{TC}_{min} is of size 19. This implies that we need to traverse all solutions of size 13–19 to find C^{TC}_{min} and there may exist a considerable number of such solutions. According to extensive experiments, a larger m leads to a larger initial C, which reduces the number of iterations. However, a larger m also increases the time for the computation of $CMIN_0$. So m is the critical parameter in our control

algorithms and has to be properly chosen. For this network, TC has the best efficiency when $m = 10$.

Finally, the numbers for $\mathsf{T}_{\mathsf{C}_{\min}^{\mathsf{FC}}}$ in Table 4 also show that our method is very efficient and scales well even for large-scale networks.

5 Conclusion

In this work, we have described a method to identify a minimal set of nodes $\mathsf{C}_{\min}$, by perturbing which, for a single time-step, the network can be driven from any initial state in a source attractor to any target attractor. This method is adapted to solve the target control and full control of large-scale BNs. Compared with the traditional methods of control where the perturbation is applied for an extended period, such a control strategy is also realistic and easier to carry out in biological lab experiments. We showed that our method is efficient and the nodes required to control the network form a small subset of the set of all nodes in the network.

In the future, as a continuation of the current work, we would like to apply our control algorithm to larger real-life biological networks and study its performance and applicability. As mentioned in Section 4, we found that the size of the partitions, m, has a big influence on the efficiency of our method. We would like to explore whether this is caused by a structural, or dynamic property of the network or a combination of the two. We would also like to extend our work to the setting of probabilistic Boolean networks (PBNs) and explore if and how to adapt the single-step control strategy to such networks and design efficient algorithms for their implementation.

Funding

This work was partially supported by the project SEC-PBN funded by University of Luxembourg and the ANR-FNR project AlgoReCell (INTER/ANR/15/11191283) funded by Luxembourg National Research Fund.

Conflict of Interest: none declared.

Acknowledgements

A.B. contributed to this work while doing an internship at the Computer Science and Communications Research Unit, University of Luxembourg.

References

Adai,A.T. *et al.* (2004) LGL: creating a map of protein function with an algorithm for visualizing very large biological networks. *J. Mol. Biol.*, **340**, 179–190.

Blanchini,F. and Franco,E. (2011) Structurally robust biological networks. *BMC Syst. Biol.*, **5**, 74.

Cervera,L. *et al.* (2015) Extended gene expression by medium exchange and repeated transient transfection for recombinant protein production enhancement. *Biotechnol. Bioeng.*, **112**, 934–946.

Charrier,S. *et al.* (2011) Quantification of lentiviral vector copy numbers in individual hematopoietic colony-forming cells shows vector dose-dependent effects on the frequency and level of transduction. *Gene Ther.*, **18**, 479–487.

Cohen,D.P.A. *et al.* (2015) Mathematical modelling of molecular pathways enabling tumour cell invasion and migration. *PLoS Comput. Biol.*, **11**, e1004571.

Conroy,B.D. *et al.* (2014) Design, assessment, and in vivo evaluation of a computational model illustrating the role of CAV1 in CD4+ T-lymphocytes. *Front. Immunol.*, **5**, 599.

Cornelius,S.P. *et al.* (2013) Realistic control of network dynamics. *Nature Communications*, **4**, 1942.

D'haeseleer,P. *et al.* (2000) Genetic network inference: from co-expression clustering to reverse engineering. *Bioinformatics*, **16**, 707–726.

Fiedler,B. *et al.* (2013) Dynamics and control at feedback vertex sets. I: informative and determining nodes in regulatory networks. *J. Dyn. Differ. Equ.*, **25**, 563–604.

Gates,A.J. and Rocha,L.M. (2016) Control of complex networks requires both structure and dynamics. *Sci. Rep.*, **6**, 24456.

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

Grieco,L. *et al.* (2013) Integrative modelling of the influence of MAPK network on cancer cell fate decision. *PLoS Comput. Biol.*, **9**, e1003286.

Hecker,M. *et al.* (2009) Gene regulatory network inference: data integration in dynamic models—a review. *BioSystems*, **96**, 86–103.

Hofherr,A. *et al.* (2017) Efficient genome editing of differentiated renal epithelial cells. *Pflugers Arch.*, **469**, 303–311.

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

Kalman,R.E. (1963) Mathematical description of linear dynamical systems. *J. SIAM*, **1**, 152–192.

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

Kim,J. *et al.* (2013) Discovery of a kernel for controlling biomolecular regulatory networks. *Sci. Rep.*, **3**, 2223.

Krumsiek,J. *et al.* (2011) Hierarchical differentiation of myeloid progenitors is encoded in the transcription factor network. *PLoS One*, **6**, e22649.

Liu,Y.-Y. *et al.* (2011) Controllability of complex networks. *Nature*, **473**, 167–173.

Mandon,H. *et al.* (2016) Relationship between the reprogramming determinants of Boolean networks and their interaction graph. In: *Proceedings of 5th International Workshop on Hybrid Systems Biology, Grenoble, France*. Vol. **9957** of LNCS. Springer, pp. 113–127.

Michels,A.J. and Frei,B. (2013) Myths, artifacts, and fatal flaws: identifying limitations and opportunities in vitamin C research. *Nutrients*, **5**, 5161–5192.

Mizera,A. *et al.* (2017) A new decomposition method for attractor detection in large synchronous Boolean networks. In: *Proceedings of 3rd International Symposium on Dependable Software Engineering: Theories, Tools, and Applications, Changsha, China*. Vol. **10606** of LNCS. Springer, pp. 232–249.

Mizera,A. *et al.* (2018) ASSA-PBN: a toolbox for probabilistic **Boolean** networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **15**, 1203–1216.

Mizera,A. *et al.* (2019) Taming asynchrony for attractor detection in large Boolean networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **16**, 31–42.

Mochizuki,A. *et al.* (2013) Dynamics and control at feedback vertex sets. II: a faithful monitor to determine the diversity of molecular activities in regulatory networks. *J. Theor. Biol.*, **335**, 130–146.

Naldi,A. *et al.* (2010) Diversity and plasticity of Th cell types predicted from regulatory network modelling. *PLoS Comput. Biol.*, **6**, e1000912.

Offermann,B. *et al.* (2016) Boolean modeling reveals the necessity of transcriptional regulation for bistability in PC12 cell differentiation. *Front. Genet.*, **7**, 44.

Paul,S. *et al.* (2018) A decomposition-based approach towards the control of Boolean networks. In: *Proceedings of 9th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics, Washington, DC, USA*. ACM Press, pp. 11–20.

Remy,E. *et al.* (2015) A modeling approach to explain mutually exclusive and co-occurring genetic alterations in bladder tumorigenesis. *Cancer Res.*, **75**, 4042–4052.

Saez-Rodriguez,J. *et al.* (2007) A logical model provides insights into T cell receptor signaling. *PLoS Comput. Biol.*, **3**, e163.

Schlatter,R. *et al.* (2009) ON/OFF and beyond—a Boolean model of apoptosis. *PLoS Comput. Biol.*, **5**, e1000595.

Singh,A. *et al.* (2012) Boolean approach to signalling pathway modelling in HGF-induced keratinocyte migration. *Bioinformatics*, **28**, 495–501.

Del Sol,A. and Buckley,N. (2014) Concise review: a population shift view of cellular reprogramming. *Stem Cells*, **32**, 1367–1372.

Tyson,J.J. *et al.* (2001) Network dynamics and cell physiology. *Nat. Rev. Mol. Cell Biol.*, **2**, 908–916.

Tyson,J.J. *et al*. (2003) Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell. *Curr. Opin. Cell Biol*., **15**, 221–231.

Wang,L.-Z. *et al*. (2016) A geometrical approach to control and controllability of nonlinear dynamical networks. *Nat. Commun*., **7**, 11323.

Wang,R.-S. *et al*. (2012) Boolean modeling in systems biology: an overview of methodology and applications. *Phys. Biol*., **9**, 055001.

Yuan,Q. *et al*. (2016) Improving BDD-based attractor detection for synchronous Boolean networks. *Sci. China Inf. Sci*., **59**, 080101.

Zañudo,J.G. and Albert,R. (2015) Cell fate reprogramming by control of intracellular network dynamics. *PLoS Comput. Biol*., **11**, e1004193.

Zañudo,J.G.T. *et al*. (2017) Structure-based control of complex networks with nonlinear dynamics. *Proc. Natl. Acad. Sci. USA*, **114**, 7234–7239.

Zhao,Y. *et al*. (2016) Control of large-scale Boolean networks via network aggregation. *IEEE Trans. Neural Netw. Learn. Syst*., **27**, 1527–1536.

Zhu,Y. *et al*. (2015) Direct conversion of human myoblasts into brown-like adipocytes by engineered super-active PPARγ. *Obesity (Silver Spring)*, **23**, 1014–1021.