Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Optimal one-bit perturbation in Boolean networks based on cascading aggregation*

Jin-feng PAN‡[1], Min MENG[2]

[1]*School of Mathematics and Information Sciences, Weifang University, Weifang 261061, China*

[2]*School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore*

E-mail: panjinfeng1989@163.com; mengminmath@gmail.com

Received Aug. 17, 2019; Revision accepted Oct. 19, 2019; Crosschecked Nov. 15, 2019

Abstract: We investigate the problem of finding optimal one-bit perturbation that maximizes the size of the basin of attractions (BOAs) of desired attractors and minimizes the size of the BOAs of undesired attractors for large-scale Boolean networks by cascading aggregation. First, via the aggregation, a necessary and sufficient condition is given to ensure the invariance of desired attractors after one-bit perturbation. Second, an algorithm is proposed to identify whether the one-bit perturbation will cause the emergence of new attractors or not. Next, the change of the size of BOAs after one-bit perturbation is provided in an algorithm. Finally, the efficiency of the proposed method is verified by a T-cell receptor network.

Key words: Large-scale Boolean network; Attractor; Cascading aggregation; One-bit perturbation
https://doi.org/10.1631/FITEE.1900411 **CLC number:** O223

1 Introduction

As a mathematical basis to model gene regulatory networks (Kauffman, 1969), Boolean networks (BNs) or probabilistic BNs have been widely studied in recent years. In BNs, each node (gene) can take two values: 1 (on) or 0 (off). Assuming the time is discrete, the value of each node at a given time is determined by a Boolean function of its neighborhood at a previous time. The vector composed of all values of nodes at a given time is called a state of the BN. Therefore, for the state of a BN with n nodes, there are 2^n values to be taken. With Boolean functions of the network, the state transition graph can be obtained.

Recently, a new matrix product, semi-tensor product of matrices, has been proposed by Cheng et al. (2011). Based on it, BNs can be converted to algebraic forms, under which many interesting results can be obtained, for example, the decoupling problem (Zhang et al., 2014; Liu Y et al., 2017; Yu et al., 2019a), pinning controllability (Lu et al., 2016), stabilization (Lu et al., 2018b), observability (Yu et al., 2019b), optimal problems (Zhu et al., 2018), control design of BNs (Liu YS et al., 2018), problems about probabilistic BNs (Liu JY et al., 2019; Wang and Feng, 2019), fuzzy relations (Fan et al., 2020), and some other problems (Ding et al., 2017; Lu et al., 2017, 2018a, 2018c; Xu et al., 2018; Zhu et al., 2018; Li and Ding, 2019; Li et al., 2019; Wang et al., 2019; Zhong et al., 2020).

From any initial state, the state of the network is limited to a subset of all possible states, called an attractor. This attractor is called a singleton attractor if it has only one state. It is called a length-l cycle if it contains l ($l \geq 2$) states. The states that flow into the same attractor make up the basin of attractions (BOAs) of the attractor. The size of the

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (No. 61773371)

ORCID: Jin-feng PAN, https://orcid.org/0000-0001-8567-3121

BOA of one attractor is the number of states in it (including the attractor itself). The larger the size of the BOA is, the more states will end up in this associate attractor.

Finding all attractors (singleton attractors or cycles) of a network and analyzing the change of the state transition network after intervention play important roles in protecting the desired attractor from being damaged. For example, genetic mutations may cause a wide variety of diseases through affecting the within-cell signaling and regulatory networks and altering the behavior of cells. The control of gene regulatory networks has been widely studied (Shmulevich et al., 2002a, 2002c). There are two kinds of control, i.e., external control and structure intervention. In this study, the one-bit perturbation considered is the simplest structure intervention. Structure intervention is to directly change the network structure (Shmulevich et al., 2002b). For example, in biology, drugs can be used to affect the state of the target genes in some specific situations. With structure intervention, the state transition graph of the whole network may change. The aim of structure intervention is to find the optimal structure intervention, under which the size of the BOAs of the desired attractors is maximized. The problem has been much investigated. Xiao and Dougherty (2007) first studied the impact of function perturbation on the BOAs of BNs based on a state transition graph. Li et al. (2012) studied the impact of function perturbation using the semi-tensor product (Cheng et al., 2011); their method is more applicable to multiple function perturbations and the cycle case. After intervention, the stabilization of a BN may be changed. In Campbell and Albert (2014), stabilization of attractors after perturbation was investigated through compensatory interactions. In Ostrowski et al. (2016), BN identification from perturbation was proposed. These methods are effective; however, when the number of nodes in the networks tends to be larger, they become inappropriate.

In Hu et al. (2016), an efficient algorithm was provided to identify the optimal one-bit perturbation; their method can be applied to networks with about 25 nodes. In Zhao et al. (2013, 2016), an algorithm was proposed to find the attractors of BNs based on network aggregation, significantly reducing the computation load. In this study, we use cascading aggregation to find the optimal one-bit perturbation for BNs with a special structure.

The problem to be solved is to find the optimal one-bit perturbation that maximizes the size of the BOAs of the desired attractors and minimizes the size of the BOAs of the undesired attractors. An effective way to deal with the problem is network aggregation. The problem can be solved in four steps. First, divide the network into several blocks. Second, check the one-bit perturbations that destroy the desired attractors and cause the emergence of new attractors. Third, find the change of the size of the BOAs of the desired attractors and undesired attractors. Fourth, pick up the optimal one-bit perturbation in the candidate perturbation set.

The main contributions of this study are as follows: network aggregation is first used to find the optimal one-bit perturbation, under which we need to consider only the perturbed subsystem to find the one-bit perturbation that causes the emergence of attractors. The problem of finding the basin of a state is partitioned to finding the basin of the corresponding state in each subnetwork, which reduces the computation load.

2 Preliminaries and problem formulation

2.1 One-bit perturbation

Consider the following BN, which contains n state nodes, $x_1, x_2, \dots, x_n$:

$$\begin{cases} x_1(t+1)=f_1\left(\left[x_j(t)\right], j\in N_1\right), \\ x_2(t+1)=f_2\left(\left[x_j(t)\right], j\in N_2\right), \\ \quad\vdots \\ x_n(t+1)=f_n\left(\left[x_j(t)\right], j\in N_n\right), \end{cases} \tag{1}$$

where $x_i(t)$ can take 1 or 0 and f_i $(i=1,2,\dots,n)$ are logical functions. $N_i\subseteq\{1,2,\dots,n\}$ is an index set, which expresses the adjacency relation of nodes corresponding to genes.

A one-bit perturbation can be accessed in Eq. (1) by choosing one Boolean function f_i $(i=1,2,\dots,n)$ and flipping the output value for a specific input of this function. For example, let f_i^j be the output value to the j^{th} entry in the truth table of f_i before perturbation. Denote the one-bit perturbation $f_i^{(j)}$ as a change in the output value of the j^{th} entry in its truth table to $1-f_i^j$. If $|N_i|=k_i$,

then the one-bit perturbation can cause 2^{n-k_i} state transition changes.

Example 1 Consider a Boolean function $f_1(x_1, x_2, x_3)$. The truth table of f_1 is given in Table 1. If a one-bit perturbation occurs on the 3rd entry of the truth table, then $f_1^3 = 1$ is converted to $1 - f_1^3 = 0$.

Table 1 The truth table of $f_1(x_1, x_2, x_3)$

Label	1	2	3	4	5	6	7	8
$x_1x_2x_3$	111	110	101	100	011	010	001	000
f_1	1	0	1	0	0	0	0	1

The states that flow into the same attractor make up the BOAs of the attractor, and the size of the BOAs of an attractor is the number of states contained in it. The size of the BOAs of a set of attractors is the number calculated by adding every size of the BOAs of each attractor.

For Eq. (1), assuming that the set of desired attractors and the set of undesired attractors are, respectively, given by

$$A_{\mathrm{d}} = \{d_1, d_2 \ldots, d_\eta\}, \tag{2}$$

$$A_{\mathrm{ud}} = \{\mathrm{ud}_1, \mathrm{ud}_2 \ldots, \mathrm{ud}_\epsilon\}. \tag{3}$$

Our aim is to find the optimal one-bit perturbation such that the following conditions hold: First, after one-bit perturbation the BOAs of the desired attractors are to be maximized and the BOAs of the undesired attractors are to be minimized. In addition, the desired attractors are not destroyed, and no new attractors are generated.

The problem can be converted to maximize the following objective function:

$$\begin{aligned} \Delta_B = \; & \textstyle\sum_{d_i \in A_{\mathrm{d}}} \left[B_j^{(p)}(d_i) - B(d_i) \right] \\ & - \textstyle\sum_{\mathrm{ud}_o \in A_{\mathrm{ud}}} \left[B_j^{(p)}(\mathrm{ud}_o) - B(\mathrm{ud}_o) \right], \end{aligned} \tag{4}$$

where $B(d_i)$ and $B_j^{(p)}(d_i)$ denote the size of the BOAs for attractor d_i before and after perturbation $f_j^{(p)}$, respectively, $i = 1, 2, \ldots, \eta$, and $j = 1, 2, , \ldots, \epsilon$. This objective function has been proposed in Hu et al. (2016).

2.2 Aggregation of Boolean networks

Denote $\chi = \{x_1, x_2, \ldots, x_n\}$. Then χ can be partitioned into s blocks $\chi = \chi_1 \cup \chi_2 \cup \ldots \cup \chi_s$, where $\chi_i \neq \varnothing, \chi_i \subset \chi$, and $\chi_i \cap \chi_j = \varnothing$ for $i \neq j$. Let $\chi_i = \{x_{i1}, x_{i2}, \ldots, x_{in_i}\}$, where n_i is the number of state nodes in the i^{th} block, and $\sum_{i=1}^{s} n_i = n$.

Denote the sets of starting nodes of the incoming and outgoing edges of block χ_i as $\mathcal{U}_i = \{u_{i1}, u_{i2}, \ldots, u_{im_i}\}$ and $\mathcal{Y}_i = \{y_{i1}, y_{i2}, \ldots, y_{ip_i}\}$, respectively. $\mathcal{U}_i$ ($\mathcal{Y}_i$) may be an empty set, when there are no incoming (outgoing) edges of block χ_i. Let C be the set of starting nodes whose edges are cut by the aggregation. Then we have

$$C = \cup_{i=1}^{s} \mathcal{U}_i = \cup_{i=1}^{s} \mathcal{Y}_i.$$

Therefore, the subnetwork Σ_i with nodes χ_i and inputs $\mathcal{U}_i$ is a Boolean control network, which can be described as

$$\begin{aligned} \Sigma_i : x_{ij}(t+1) = & f_{ij}\left(x_{i1}(t), x_{i2}(t), \ldots, x_{in_i}(t), \right. \\ & \left. u_{i1}(t), u_{i2}(t), \ldots, u_{im_i}(t)\right) \end{aligned} \tag{5}$$

for $i = 1, 2, \ldots, s$ and $j = 1, 2, \ldots, n_i$.

Definition 1 (Zhao et al., 2016) The aggregation is said to be cascading if (with possible reordering) the groups

$$\chi^i = \cup_{j=q(1)}^{q(i)} \chi_j \ (i = 1, 2, \ldots, s) \tag{6}$$

have zero in-degree, where $q(\cdot)$ is a one-to-one mapping from $\{1, 2 \ldots, s\}$ to itself.

Definition 2 (Zhao et al., 2013) The aggregation is said to be acyclic if its aggregation graph contains no cycle.

Lemma 1 (Klamt et al., 2006) An aggregation is a cascading aggregation if and only if it is an acyclic aggregation.

The method to find an acyclic aggregation of a network has been given in Zhao et al. (2013). In this study, we consider only the BNs that can be partitioned by cascading aggregation.

Example 2 Consider the BN in Fig. 1, which contains nine nodes. The node set is $\chi = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9\}$ and can be partitioned into three blocks χ_i ($i = 1, 2, 3$), where $\chi_1 = \{x_1, x_2, x_3\}$, $\chi_2 = \{x_4, x_5\}$, and $\chi_3 = \{x_6, x_7, x_8, x_9\}$. Reorder the blocks, and define $\chi^1 = \chi_3$, $\chi^2 = \chi_3 \cup \chi_1$, and $\chi^3 = \chi$. By Definition 1, the aggregation is cascading.

The problem to be solved is analyzed based on the state transition graph of BNs. So, we present the definition of the state transition graph of a BN first.

Definition 3 The state transition graph of a BN is an ordered pair $G = (V, E)$, where V denotes a set

of vertices and each vertex is a state. A BN with n nodes has 2^n vertices. Let E denote a set of edges. If $f(v_j) = v_i$, where $f : \Delta^n \to \Delta^n$ is the Boolean function of this network, we have $(v_j, v_i) \in E$.

In this study, the sizes of the basins of states (BOSs) are used to calculate Δ_B. To find the basin of states, the definitions of the immediate predecessor (successor) and predecessor (successor) of a given vertex are provided as follows:

Definition 4 (Liu M, 2015) The immediate predecessor and immediate successor sets of a vertex $v \in V$ in a directed graph are defined as $\text{ipred}(v) = \{u|(u, v) \in E\}$ and $\text{isucc}(v) = \{u|(v, u) \in E\}$, respectively.

Definition 5 (Liu M, 2015) The predecessor set $\text{pred}(v)$ of a vertex $v \in V$ is a subset of V containing all vertices in which v is reachable. Similarly, the successor set $\text{succ}(v)$ of a vertex $v \in V$ is a subset of V containing all vertices reachable from v.

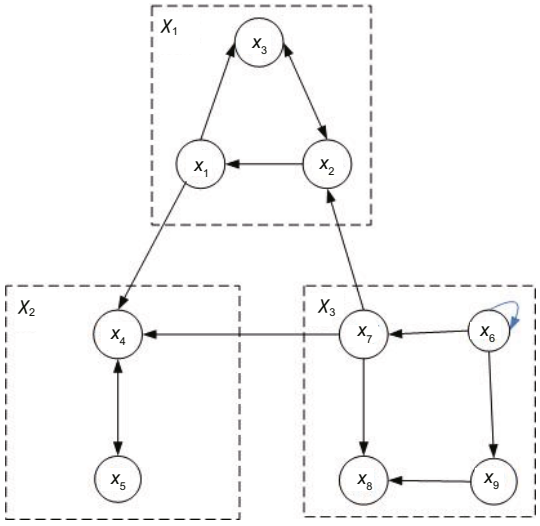

Fig. 1 Aggregation of a network with nine nodes into three Boolean control networks

3 Main results

In this study, the one-bit perturbation considered must satisfy the following rules:

1. After one-bit perturbation, desired attractors cannot be destroyed.

2. No new attractors are generated.

Therefore, we need to identify such one-bit perturbations first.

Lemma 2 Assume that $A = \{a_1 \to a_2 \to \ldots \to a_k \to a_1\}$ is a length-k attractor of Eq. (1). A is not affected by the one-bit perturbation if and only if after the perturbation, a_i is still mapped to a_{i+1} (when $i = k$, let $k + 1 := 1$).

To ensure the invariance of the desired attractors, the states in the desired attractors cannot be perturbed.

3.1 Revealing attractors of subnetworks

If Eq. (1) is partitioned into $\{\chi_1, \chi_2, \ldots, \chi_s\}$, the subnetwork Σ_i is described as Eq. (5). Define $\Psi(\cdot, S)$ as a projection of state a onto state $b \in S$ ($l = |S|$), with order $(x_{i_1}, x_{i_2}, \ldots x_{i_l})$ and $i_1 < i_2 < \ldots < i_l$. Define $\phi(\cdot, \Sigma_i) : D^n \to D^{n_i+m_i}$ as the projection of state a_1 in D^n onto state $b_1 \in D^{n_i+m_i}$ with order $(x_{\kappa_1}, x_{\kappa_2}, \ldots, x_{\kappa_{m_i}}, x_{\nu_1}, x_{\nu_2}, \ldots, x_{\nu_{n_i}})$ and $\kappa_1 < \kappa_2 < \ldots < \kappa_{m_i}, \nu_1 < \nu_2 < \ldots < \nu_{n_i}$. Here, x_{κ_i} $(i = 1, 2, \ldots, m_i)$ are the inputs of Σ_i. Define projection $\Phi(\cdot, \Sigma_i)$ as the projection of attractor a onto a path in Σ_i: for a length-k attractor $a = \{a_1 \to a_2 \to \ldots \to a_k \to a_1\}$, the projection $\Phi(\cdot, \Sigma_i)$ is given by

$$\begin{aligned}\Phi(a, \Sigma_i) := \{&\phi(a_1, \Sigma_i) \to \phi(a_2, \Sigma_i) \to \ldots \\ &\to \phi(a_k, \Sigma_i) \to \phi(a_1, \Sigma_i)\}.\end{aligned}$$

Example 3 Considering Example 2 again, assume that a length-4 attractor is given as

$$\begin{aligned}A = \{&(1,0,0,1,0,0,1,1,0) \to (0,0,0,1,0,1,0,1,1) \\ &\to (0,0,1,1,0,0,1,1,0) \to (0,1,1,1,1,1,0,1,1) \\ &\to (1,0,0,1,0,0,1,1,0)\},\end{aligned}$$

where the state is ordered as $(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9)$. Then the length-4 path in Σ_1 can be obtained by projection

$$\begin{aligned}\Phi(A, \Sigma_1) = \{&(1,1,0,0) \to (0,0,0,0) \to (1,0,0,1) \\ &\to (0,0,1,1) \to (1,1,0,0)\},\end{aligned}$$

the length-4 path in Σ_2 can be obtained by projection

$$\begin{aligned}\Phi(A, \Sigma_2) = \{&(1,1,1,0) \to (0,0,1,0) \to (0,1,1,0) \\ &\to (0,0,1,1) \to (1,1,1,0)\},\end{aligned}$$

and the length-4 path in Σ_3 can be obtained by projection

$$\begin{aligned}\Phi(A, \Sigma_3) = \{&(0,1,1,0) \to (1,0,1,1) \to (0,1,1,0) \\ &\to (1,0,1,1) \to (0,1,1,0)\};\end{aligned}$$

i.e., the input-state cycle in Σ_3 is

$$\{(0,1,1,0)\to(1,0,1,1)\to(0,1,1,0)\}.$$

Theorem 1 For Eq. (1), the desired attractors (Eq. (2)) stay the same after a one-bit perturbation, if and only if for subsystems $\Sigma_1,\Sigma_2,\dots,\Sigma_s$, cycles $\Phi(d_{r_1},\Sigma_{r_2})$ remain unchanged ($r_1\in\{1,2,\dots,\eta\}$ and $r_2\in\{1,2,\dots,s\}$).

Example 4 Consider Example 3 again. Assume that

$$\begin{aligned}A_1=&\{(1,1,1,1,1,1,1,0,0)\to(1,0,1,1,1,1,1,1,1)\\&\to(1,1,1,1,1,1,1,1,0)\to(1,0,1,1,1,1,1,1,0)\\&\to(1,1,1,1,1,1,1,0,0)\}\end{aligned}$$

is the desired attractor. Using Theorem 1, for subsystem Σ_3, $f_6(0)$, $f_7(1)$, $f_8(1,0)$, and $f_9(0)$ should remain unchanged; i.e., $f_6^{(2)}$, $f_7^{(1)}$, $f_8^{(2)}$, and $f_9^{(2)}$ are not the one-bit perturbation candidates.

Similarly, for subsystem Σ_1, $f_1^{(2)}$, $f_2^{(3)}$, and $f_3^{(4)}$ are not the one-bit perturbation candidates. For subsystem Σ_2, $f_4^{(7)}$ and $f_5^{(1)}$ are not the one-bit perturbation candidates.

Without loss of generality, in this study we assume that $\chi_1,\chi_2,\dots,\chi_s$ satisfy Eq. (6).

3.2 Algorithm for finding the basin of states

Define the BOS size of state a as the number of transient states (written as BOS(a)) that can reach a plus 1 (i.e., $|\text{pred}(a)|+1$). For example, in Fig. 2, the BOS of state 0101 before perturbation is $\{1110,0010,1001,0000,1010,0111,1000\}$, so BOS(0101) = 8.

The state transition matrix of a BN with n nodes is constructed as follows: First, the size of this matrix is $2^n\times 2$. For the i^{th} row ($i=1,2,\dots,2^n$), the first element is $i-1$, and its corresponding binary number is written as dec2bin($i-1$). The second element is the decimal number corresponding to the image of dec2bin($i-1$).

The input-state transition matrix of a Boolean control network with n nodes and m inputs is defined based on the following conditions:

1. The size is $2^{m+n}\times 3$.

2. For the i^{th} row ($i=1,2,\dots,2^{m+n}$), the first element is $\text{mod}(i,2^n)-1$ (if $\text{mod}(i,2^n)=0$, the first element is 2^n-1). The second element is the decimal number corresponding to the image with input dec2bin(floor($i/2^m$)) and state dec2bin(rem($i/2^m$)),

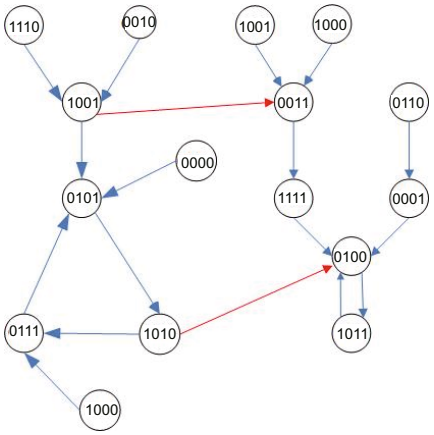

Fig. 2 A simple Boolean network with four nodes. The changed arrows after perturbation are marked by the red color

References to color refer to the online version of this figure

where rem($i/2^m$) is the reminder of $i/2^m$, and floor($i/2^m$) is the divisor of $i/2^m$. The third element is floor($i/2^m$).

For ease of presentation, we first give some notations. For matrix $\boldsymbol{A}$, size($\boldsymbol{A}$, 1) is the number of rows in $\boldsymbol{A}$ and size($\boldsymbol{A}$, 2) is the number of columns in $\boldsymbol{A}$. $\Sigma_1*\Sigma_2*\ldots*\Sigma_k$ is the subnetwork composed of $\Sigma_1,\Sigma_2,\dots,\Sigma_k$. The state transition matrix of Σ_1 is denoted by $\boldsymbol{B}_1$, and the input-state transition matrices of $\Sigma_2,\Sigma_3,\dots,\Sigma_s$ are denoted by $\boldsymbol{B}_2,\boldsymbol{B}_3,\dots,\boldsymbol{B}_s$, respectively.

Assume $\boldsymbol{c}=[c_1,c_2,\dots,c_m]\in\mathbb{R}^{1\times m}$, $\boldsymbol{d}=[d_1,d_2,\dots,d_n]\in\mathbb{R}^{1\times n}$, and $c_i,d_j\in\mathbb{R}$. Define $\bar{*}$ as follows: $\boldsymbol{c}\bar{*}\boldsymbol{d}=[(c_1,d_1),(c_1,d_2),\dots,(c_1,d_n),\dots,(c_m,d_n)]$. Given two row vectors $\boldsymbol{C}$ and $\boldsymbol{D}$, $\boldsymbol{C}=[\boldsymbol{C}_1,\boldsymbol{C}_2,\dots,\boldsymbol{C}_p]$ and $\boldsymbol{D}=[\boldsymbol{D}_1,\boldsymbol{D}_2,\dots,\boldsymbol{D}_p]$, the matrix product $\bar{\otimes}$ of $\boldsymbol{C}$ and $\boldsymbol{D}$ is defined as $\boldsymbol{C}\bar{\otimes}\boldsymbol{D}=[\boldsymbol{C}_1\bar{*}\boldsymbol{D}_1,\boldsymbol{C}_2\bar{*}\boldsymbol{D}_2,\dots,\boldsymbol{C}_p\bar{*}\boldsymbol{D}_p]$.

To obtain the basin of state a, we need to find ipred(a) first. The algorithm to seek ipred(a) is presented in Algorithm 1.

For each state in $\boldsymbol{V}^{s^1}$, repeating the process presented above, we can obtain the neighbors that can reach a in two steps. Here, we use $\boldsymbol{V}^{s^2}$ to denote their set. Continue the process until $|\boldsymbol{V}^{s^k}|>0$ and $|\boldsymbol{V}^{s^{k+1}}|=0$. Then the basin of state a is derived, i.e., $\bigcup_{i=1}^{k}\boldsymbol{V}^{s^i}$.

3.3 Emergence of new attractors

In this study, new attractors are not wished to emerge, so we need to find such one-bit perturbations

and avoid using them. We discuss the problem in two conditions. One case is the one-bit perturbation that occurs in the root block, and the other is the one-bit perturbation that occurs in blocks with inputs.

When one-bit perturbation occurs in the root block, we have the following result:

Corollary 1 The emergence of new attractors in the root block will cause the emergence of new attractors for the whole BN.

When one-bit perturbation emerges in block Σ_i with inputs, attractors in $\Sigma_1 * \Sigma_2 * \ldots * \Sigma_{i-1}$ are not destroyed; i.e., the input sequences corresponding to attractors in $\Sigma_1 * \Sigma_2 * \ldots * \Sigma_{i-1}$ are not changed. Therefore, a new attractor emerges for the whole network when one of the following two conditions is satisfied:

1. The one-bit perturbation appears on the input-state cycles of Σ_i.

2. Under the input sequences corresponding to attractors in $\Sigma_1 * \Sigma_2 * \ldots * \Sigma_{i-1}$, a new cycle emerges after the one-bit perturbation.

Two algorithms are given to judge whether the one-bit perturbation causes the emergence of new attractors. First, we consider the one-bit perturbation that occurs in root block Σ_1. Presenting the original attractors of Σ_1 in a column matrix $\tilde{\boldsymbol{A}}$, the state transition matrix after a one-bit perturbation is written as $\bar{\boldsymbol{A}}$, and the perturbed states in Σ_1 constitute a row matrix $\boldsymbol{A}_1$. Algorithm 2 identifies the emergence of new attractors in the root block.

If the one-bit perturbation does not occur in the root block, assume the one-bit perturbation occurs in Σ_e.

Write the input-state transition matrix as $\hat{\boldsymbol{B}}$ after one-bit perturbation, and assume the number of attractors in $\Sigma_1 * \Sigma_2 * \ldots * \Sigma_{e-1}$ is ξ. Suppose one attractor in system $\Sigma_1 * \Sigma_2 * \ldots * \Sigma_{e-1}$ is $\boldsymbol{A}_i$. In Σ_e, assume the number of input-state cycles corresponding to $\boldsymbol{A}_i$ is k_i, and the longest cycle is of length l_i. For all input-state cycles, expand their lengths to l_i. Denote matrix $\boldsymbol{C}_i \in \mathbb{R}^{l_i \times (k_i+1)}$ as follows: Taking the input sequence of the longest cycle as the last column of $\boldsymbol{C}_i$, the other columns are the states of the corresponding input-state cycles in Σ_e. Construct matrix $\bar{\boldsymbol{B}}$ by choosing the perturbed states in $\hat{\boldsymbol{B}}$.

Based on the above representation, Algorithm 3 is proposed to identify the emergence of new attractors in blocks with inputs.

3.4 $\Delta_{\boldsymbol{B}}$ after one-bit perturbation

After one-bit perturbation, if the perturbed states still enter the original desired attractors or the undesired attractors, this kind of perturbed state

Algorithm 1 Seeking ipred(a)

Input: Set $\Omega = \{1, 2, \ldots, s\}$, $V_i = [\,], i \in \Omega$, $c_i = 0$
Output: $\boldsymbol{V}^{s^1}$ // ipred(a)
for all $k \in \Omega$ **do**
  $s_k \leftarrow \Psi(a, \chi_k)$
  for $i_1 = 0$ to size($\boldsymbol{B}_1, 1$) **do**
    if $\boldsymbol{B}_1(i_1, 2) = s_1$ **then**
      $\boldsymbol{V}_1[c_1 + +] \leftarrow \boldsymbol{B}_1(i_1, 1)$
    end if
  end for
  $\boldsymbol{V}_1 \leftarrow [V_{11}, V_{12}, \ldots, V_{1(2^{m_2})}]$ //$\Psi(V_{1j}, \mathcal{U}_2) = j - 1$ // $(j = 1, 2, \ldots, 2^{m_2})$
end for
for all $l \in \Omega \backslash \{1\}$ **do**
  for $i_l = 1$ to size($\boldsymbol{B}_l, 1$) **do**
    for $r_l = 1$ to 2^{m_l} **do**
      if $\boldsymbol{B}_l(i_l, 2) = s_l$ **and** $\boldsymbol{B}_l(i_l, 3) = r_l - 1$ **then**
        $\boldsymbol{V}_{lr_1}[c_l + +] \leftarrow \boldsymbol{B}_l(i_l, 1)$
      end if
      $\boldsymbol{V}_l \leftarrow [V_{l1}, V_{l2}, \ldots, V_{l(2^{m_l})}]$
      $\boldsymbol{V}^{1^1} \leftarrow \boldsymbol{V}_1$
      $\boldsymbol{V}^{l^1} \leftarrow \boldsymbol{V}^{(l-1)^1} \bar{\otimes} \boldsymbol{V}_l$
    end for
  end for
end for

Algorithm 2 Identifying new attractors in the root block

Input: Set $\boldsymbol{B}_1 = [\,]$, $\boldsymbol{C}_1 = [\,]$, $R = 0$
Output: R
for $i = 0$ to size($\boldsymbol{A}_1, 2$) **do**
  for $j = 0$ to size($\bar{\boldsymbol{A}}, 1$) **do**
    $\boldsymbol{B}_1(1, i) \leftarrow \boldsymbol{A}_1(1, i)$ and
    $\boldsymbol{B}_1(j + 1, i) \leftarrow \bar{\boldsymbol{A}}(\boldsymbol{B}_1(j, i) + 1, 2)$
    for $k = 1$ to j **do**
      if $\boldsymbol{B}_1(j + 1, i) = \boldsymbol{B}_1(k, i)$ **then**
        $\boldsymbol{C}_1(i) \leftarrow \boldsymbol{B}_1(k, i)$
      end if
    end for
  end for
end for
if $\boldsymbol{C}_1 \cap \widetilde{\boldsymbol{A}} = \varnothing$ **then**
  $R \leftarrow 0$ // No new attractors emerge
else
  $R \leftarrow 1$ // The one-bit perturbation causes the // emergence of new attractors
end if

Algorithm 3 Identifying new attractors in blocks with inputs

Input: Set $\boldsymbol{B}_2 = \text{cell}(1, \xi)$, $R = 0$
Output: R

for $i = 1$ to size$(\bar{\boldsymbol{B}}, 1)$ **do**
  for $j = 1$ to ξ **do**
    for $l = 2$ to size$(\hat{\boldsymbol{B}}, 1)/2^{m_{e-1}} - 1$ **do**
      if $\bar{\boldsymbol{B}}(i, 3) \cap \boldsymbol{C}_j(, k_j + 1) = s_{ij}$ **then**
        $\boldsymbol{B}_2\{1, j\}(1, i) \leftarrow \bar{\boldsymbol{B}}(i, 1)$
        $\boldsymbol{B}_2\{1, j\}(2, i) \leftarrow \bar{\boldsymbol{B}}(i, 2)$
        $\boldsymbol{B}_2\{1, j\}(l + 1, i) \leftarrow \hat{\boldsymbol{B}}(\boldsymbol{B}_2\{1, j\}(l, i) \cdot 2^{m_{e-1}} + \boldsymbol{C}_j(l, k_j + 1) + 1, 2)$
        if there exists p such that $\{\boldsymbol{B}_2\{1, j\}(p \cdot l_i + s_{ij}, i), \boldsymbol{B}_2\{1, j\}(p \cdot l_i + s_{ij} + 1, i), \ldots, \boldsymbol{B}_2\{1, j\}((p+1) \cdot l_i + s_{ij}, i)\} \cap \{\boldsymbol{C}_j(, 1), \boldsymbol{C}_j(, 2), \ldots, \boldsymbol{C}_j(, k_j)\} = \varnothing$ **then**
          $R \leftarrow 0$ // No new attractors emerge
        else
          $R \leftarrow 1$ // The one-bit perturbation causes // the emergence of new attractors
        end if
      end if
    end for
  end for
end for

is not taken into consideration. We provide an algorithm to determine the attractors that the perturbed states will eventually enter.

If the one-bit perturbation occurs in the root block, the algorithm is similar to Algorithm 2.

If the one-bit perturbation occurs in other blocks, Algorithm 4 is presented to find the attractor that perturbed state a will eventually enter.

Using Algorithm 4, we just need to consider the perturbed states entering the desired attractors (undesired attractors) before perturbation. After perturbation, the perturbed states enter the undesired attractors (desired attractor).

For state a, $\text{pred}(a) \cup \{a\}$ is the subset of $\text{isucc}(a)$. If state a is a perturbed state, then $\text{pred}(a) \cup \{a\}$ expects the other perturbed states and the BOS of them will be the subset of $\text{isucc}(a)$ in the new state transition graph after one-bit perturbation.

For example, in Fig. 2, there are two attractors $A_1 = \{0101 \rightarrow 1010 \rightarrow 0111 \rightarrow 0101\}$ and $A_2 = \{0100 \rightarrow 1011 \rightarrow 0100\}$. States 1011 and 1010 are the perturbed states; before perturbation, isucc(1001) =0101 with BOS(1001)=3 and isucc(1010) =0101 with BOS(1010)=8. After perturbation, isucc(1001) =0011 and isucc(1010)=0100. Because state 0011 is an element of the BOA of A_2 and state 1001 is an element of the BOS of state 1010, the size of BOA of attractor A_2 is increased by five.

Algorithm 4 Finding the corresponding attractor for a given state a

1: Find the corresponding attractor for state $\Phi(a, \Sigma_1)$. Put the transition states into a row vector $\boldsymbol{D}_1$, which includes the attractor states
2: For each state in $\boldsymbol{D}_1$, find the corresponding inputs to Σ_2 and put the corresponding states into a row vector $\boldsymbol{D}_{12}$
3: $\boldsymbol{D}_2$ is the set of states in Σ_2 with $\boldsymbol{D}_2(1) = \Psi(a, \chi_2)$, and the $(j+1)^{\text{th}}$ element of $\boldsymbol{D}_2$ is determined by $\boldsymbol{D}_1(j)$ and $\boldsymbol{D}_{12}(j)$
4: $[\boldsymbol{D}_{12}, \boldsymbol{D}_2]$ does not contain input-state cycles, enlarging $\boldsymbol{D}_1$ and $\boldsymbol{D}_{12}$ with the multiple of the length of attractor until $[\boldsymbol{D}_1, \boldsymbol{D}_{12}]$ has an input-state cycle
5: Regard system $\Sigma_1 * \Sigma_2$ as the new root block, and construct matrix $\boldsymbol{D}_3$ via the similar method
6: Construct matrices $\boldsymbol{D}_4, \boldsymbol{D}_5, \ldots, \boldsymbol{D}_s$ in a similar way
7: Choosing the last elements of matrices $\boldsymbol{D}_1$, $\boldsymbol{D}_2$, …, $\boldsymbol{D}_s$, and arranging them in sequence to form a row vector, the attractor that state a will eventually enter can be found

Through the above analysis, Δ_B after one-bit perturbation can be obtained through Algorithm 5.

3.5 Method for finding the optimal one-bit perturbation

Through the above analysis, we can use the following method to find the optimal one-bit perturbation:

Step 1: Find the cascading aggregation that partitions the network into several blocks.

Step 2: Use Theorem 1 to find the one-bit perturbations that destroy the desired attractors.

Step 3: Check the one-bit perturbations that cause the emergence of new attractors based on Algorithms 2 and 3.

Step 4: On the basis of Algorithm 4, find the perturbed states entering different sets of attractors after one-bit perturbation.

Step 5: For each perturbed state obtained in step 4, calculate Δ_B based on Algorithm 5.

Step 6: Seek out the largest Δ_B, and the corresponding one-bit perturbation is the optimal one-bit perturbation.

Remark 1 Enlarge all desired attractors of the same length, and write attractor d_j as d'_j. If $\Phi(d'_1,\Sigma_i)\neq\Phi(d'_2,\Sigma_i)\neq\ldots\neq\Phi(d'_\eta,\Sigma_i)$, $i=1,2,\ldots,s$, then we just need to consider the one-bit perturbations occurring in the root block and the one-bit perturbation that causes the disappearance of undesired attractors. If the condition is not satisfied, find the first k such that at least two elements in $\{\Phi(d'_1,\Sigma_k),\Phi(d'_2,\Sigma_k),\ldots,\Phi(d'_\eta,\Sigma_k)\}$ are different. Then we just need to consider the one-bit perturbation occurring in $\Sigma_1,\Sigma_2,\ldots,\Sigma_k$, and the one-bit perturbation that causes the disappearance of undesired attractors.

Algorithm 5 Calculating Δ_B after one-bit perturbation

Input: Set $\boldsymbol{C}=[c_1,c_2,\ldots,c_l]$ // the perturbed states // that enter different attractors after the one-bit // perturbation; assuming that the maximum // distance between state c_i and the states that can // eventually reach it is k_i

Output: Δ_B

for $i=1$ to l **do**
  $C_{i1}\leftarrow \text{ipred}(c_i)/\boldsymbol{C}$
  for $j=2$ to k_i **do**
    $C_{ij}=\text{ipred}(C_{i(j-1)})/\boldsymbol{C}$ // C_{ij} is the difference // between the set of the immediate // predecessors of each state in $C_{i(j-1)}$ and $\boldsymbol{C}$
  end for
  if c_i enters a desired attractor before perturbation, and eventually enters an undesired attractor after one-bit perturbation **then**
    $\Delta_{B_i}\leftarrow -\sum_{t=1}^{k_i}|C_{it}|$
  end if
  if c_i enters an undesired attractor before perturbation, and eventually enters a desired attractor after one-bit perturbation **then**
    $\Delta_{B_i}\leftarrow \sum_{j=1}^{k_i}|C_{ij}|$
  end if
  $\Delta_B\leftarrow\sum_{i=1}^{l}\Delta_{B_i}$
end for

4 Application to bimolecular networks

In this section, the BN model of T-cell receptor kinetics (Klamt et al., 2006) is to be analyzed. Its network graph is shown in Fig. 3. For the BN, there are three inputs, CD45, CD8, and TCRlig. We fix their values to $(1,1,1)$. We use x_i to represent the nodes of this network (Fig. 4). The network is partitioned into five blocks. Its state equation is given in Eq. (7).

The attractors of Eq. (7) are $\boldsymbol{A}_1=(204,0,0,2,0)\rightarrow(204,0,0,2,0)$ and $\boldsymbol{A}_2=(221,32,0,2,130)\rightarrow(206,19,0,2,256)\rightarrow(200,8,0,2,576)\rightarrow(76,0,0,2,32)\rightarrow(132,0,0,2,16)\rightarrow(236,0,0,2,8)\rightarrow(221,32,0,2,130)$, where each state is written as a 1×5 vector and the i^{th} element is the decimal number corresponding to the binary number in χ_i. Assume that $\boldsymbol{A}_1$ is the undesired attractor and $\boldsymbol{A}_2$ is the desired attractor. First, the desired attractors should remain unchanged; thus, one-bit perturbations $f_1^{(1,3,4)}$, $f_2^{(1,3,4,7)}$, $f_3^{(1,2)}$, $f_4^{(2,3,4,8)}$, $f_5^{(1,2,3)}$, $f_6^{(1,2)}$, $f_7^{(1,2)}$, $f_8^{(1,2)}$, $f_9^{(1,2)}$, $f_{10}^{(1,2)}$, $f_{11}^{(1,2)}$, $f_{12}^{(2)}$, $f_{13}^{(1,2)}$, $f_{14}^{(1,2)}$, $f_{15}^{(8,28,31,32)}$, $f_{16}^{(2)}$, $f_{17}^{(2)}$, $f_{18}^{(2)}$, $f_{19}^{(2)}$, $f_{20}^{(2)}$, $f_{21}^{(2)}$, $f_{22}^{(2)}$, $f_{23}^{(2)}$, $f_{24}^{(2)}$, $f_{25}^{(2)}$, $f_{26}^{(2)}$, $f_{27}^{(1)}$, $f_{28}^{(1,2)}$, $f_{29}^{(1,2)}$, $f_{30}^{(1,2)}$, $f_{31}^{(2,4)}$, $f_{32}^{(1,2)}$, $f_{33}^{(1,2)}$, $f_{34}^{(1,2)}$, $f_{35}^{(1,4)}$, $f_{36}^{(1,2)}$, and $f_{37}^{(3,4)}$ should stay the same. Attractors $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ satisfy the condition in Remark 1. So, we just need to consider the one-bit perturbations in Σ_1, i.e., $f_1^{(2)}$, $f_2^{(2,5,6,8)}$, $f_4^{(1,5,6,7)}$, and $f_5^{(4)}$. Using Algorithm 2, we can see that one-bit perturbations $f_1^{(2)}$ and $f_4^{(1)}$ cause the emergence of new attractors. Using Algorithm 5, we can obtain that $f_4^{(6)}$ is the optimal

$$
\begin{cases}
x_1(t+1)=x_3(t)\vee x_6(t),\ x_2(t+1)=x_1(t)\vee(x_3(t)\wedge x_6(t)),\ x_3(t+1)=\neg x_5(t),\\
x_4(t+1)=\neg x_7(t)\wedge x_3(t)\wedge x_2(t),\ x_5(t+1)=x_1(t)\vee\neg x_6(t),\ x_6(t+1)=\neg x_7(t),\\
x_7(t+1)=x_4(t),\ x_8(t+1)=x_3(t),\ x_9(t+1)=x_4(t),\ x_{10}(t+1)=x_9(t),\\
x_{11}(t+1)=x_{10}(t),\ x_{12}(t+1)=x_{11}(t)\wedge x_4(t),\ x_{13}(t+1)=x_9(t),\ x_{14}(t+1)=x_9(t),\\
x_{15}(t+1)=(x_8(t)\vee x_{12}(t))\wedge(x_4(t)\wedge x_{11}(t)\wedge x_{14}(t)),\ x_{16}(t+1)=x_{15}(t),\\
x_{17}(t+1)=x_{16}(t),\ x_{18}(t+1)=x_{17}(t),\ x_{19}(t+1)=x_{18}(t),\ x_{20}(t+1)=x_{15}(t),\\
x_{21}(t+1)=x_{23}(t),\ x_{22}(t+1)=x_{21}(t),\ x_{23}(t+1)=x_{20}(t),\ x_{24}(t+1)=x_{22}(t),\\
x_{25}(t+1)=x_{23}(t),\ x_{26}(t+1)=\neg x_{25}(t),\ x_{27}(t+1)=\neg x_{26}(t),\ x_{28}(t+1)=x_{29}(t),\\
x_{29}(t+1)=x_{30}(t),\ x_{30}(t+1)=x_{34}(t),\ x_{31}(t+1)=x_{13}(t)\vee x_{35}(t),\ x_{32}(t+1)=x_{31}(t),\\
x_{33}(t+1)=x_{32}(t),\ x_{34}(t+1)=x_{33}(t),\ x_{35}(t+1)=x_{20}(t)\wedge x_{23}(t),\\
x_{36}(t+1)=x_{34}(t),\ x_{37}(t+1)=x_{36}(t)\wedge x_{24}(t).
\end{cases}
\tag{7}
$$

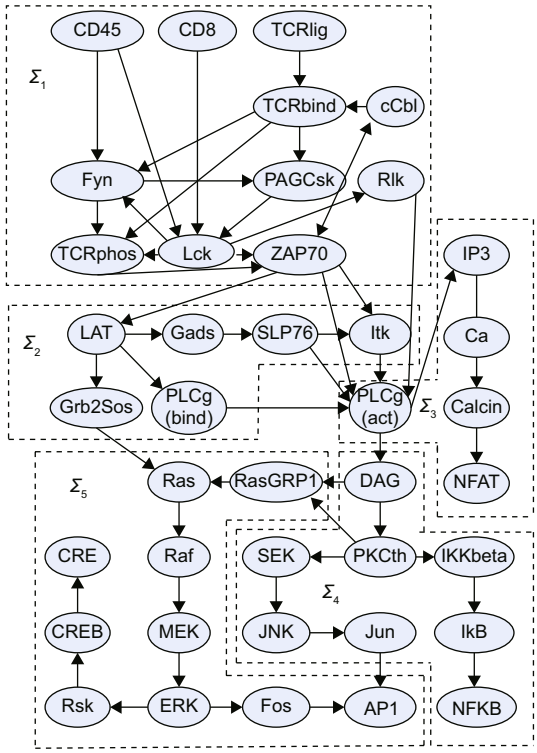

Fig. 3 A Boolean network implementing a T-cell receptor model. Adapted from Klamt et al. (2006)

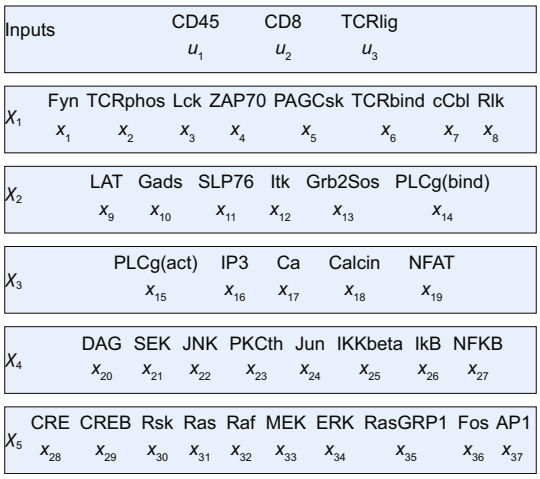

Fig. 4 Network after aggregation

one-bit perturbation.

If we use the method presented in Hu et al. (2016) to solve the problem, we need to consider all one-bit perturbation candidates for the whole network. Here, after network aggregation, we just need to consider the one-bit perturbations occurring in Σ_1, which clearly reduces the computation load.

5 Conclusions

In this study, we have considered the problem of finding the optimal one-bit perturbation of large-scale BNs. The study has been carried on the basis that the BN can be partitioned by network aggregation. After aggregation, the problem has been reduced to finding the optimal one-bit perturbation for the corresponding subnetworks. Methods to check the candidate one-bit perturbations have been given by Algorithms 2 and 3. After that, the way to find the optimal one-bit perturbation has been proposed based on Algorithms 4 and 5.

For a BN that can be partitioned through cascading aggregation, the computational complexity of finding the BOA of an attractor is $O(2^{n_1\cdot k_1}) + O(2^{n_2\cdot k_2}) + \ldots + O(2^{n_s\cdot k_s})$. Under traditional methods, the computational complexity is $O(2^{(n_1+n_2+\ldots+n_s)\cdot k})$, $k \geq k_i,\ i = 1, 2, \ldots, s$. It is clear that the computation load has been reduced significantly. For BNs that cannot be partitioned, the method proposed in our study is useless. Future work will focus on finding methods to solve the optimal intervention problem for large-scale BNs without a special structure.

Contributors

Jin-feng PAN designed the research, processed the data, and drafted the manuscript. Min MENG helped organize the manuscript. Jin-feng PAN revised and finalized the manuscript.

Compliance with ethics guidelines

Jin-feng PAN and Min MENG declare that they have no conflict of interest.

References

Campbell C, Albert R, 2014. Stabilization of perturbed Boolean network attractors through compensatory interactions. *BMC Syst Biol*, 8:53. https://doi.org/10.1186/1752-0509-8-53

Cheng DZ, Qi HS, Li ZQ, 2011. Analysis and Control of Boolean Networks: a Semi-tensor Product Approach. Springer, London, UK. https://doi.org/10.1007/978-0-85729-097-7

Ding XY, Li HT, Yang QQ, et al., 2017. Stochastic stability and stabilization of n-person random evolutionary Boolean games. *Appl Math Comput*, 306:1-12. https://doi.org/10.1016/j.amc.2017.02.020

Fan HB, Feng JE, Meng M, et al., 2020. General decomposition of fuzzy relations: semi-tensor product approach. *Fuzzy Sets Syst*, 384:75-90. https://doi.org/10.1016/j.fss.2018.12.012

Hu MX, Shen LZ, Zan XZ, et al., 2016. An efficient algorithm to identify the optimal one-bit perturbation based on

the basin-of-state size of Boolean networks. *Sci Rep*, 6:26247. https://doi.org/10.1038/srep26247

Kauffman SA, 1969. Metabolic stability and epigenesis in randomly constructed genetic nets. *J Theor Biol*, 22(3):437-467. https://doi.org/10.1016/0022-5193(69)90015-0

Klamt S, Saez-Rodriguez J, Lindquist JA, et al., 2006. A methodology for the structural and functional analysis of signaling and regulatory networks. *BMC Bioinform*, 7:56. https://doi.org/10.1186/1471-2105-7-56

Li HT, Ding XY, 2019. A control Lyapunov function approach to feedback stabilization of logical control networks. *SIAM J Contr Optim*, 57(2):810-831. https://doi.org/10.1137/18M1170443

Li HT, Wang YZ, Liu ZB, 2012. Function perturbation impact on the topological structure of Boolean networks. Proc 10th World Congress on Intelligent Control and Automation, p.1241-1246. https://doi.org/10.1109/WCICA.2012.6358071

Li HT, Xu XJ, Ding XY, 2019. Finite-time stability analysis of stochastic switched Boolean networks with impulsive effect. *Appl Math Comput*, 347:557-565. https://doi.org/10.1016/j.amc.2018.11.018

Liu JY, Liu Y, Guo YQ, et al., 2019. Sampled-data state-feedback stabilization of probabilistic Boolean control networks: a control Lyapunov function approach. *IEEE Trans Cybern*, in press. https://doi.org/10.1109/TCYB.2019.2932914

Liu M, 2015. Analysis and Synthesis of Boolean Networks. Licentiate Thesis, KTH School of Information and Communication Technology, Sweden.

Liu Y, Li BW, Lu JQ, et al., 2017. Pinning control for the disturbance decoupling problem of Boolean networks. *IEEE Trans Autom Contr*, 62(12):6595-6601. https://doi.org/10.1109/TAC.2017.2715181

Liu YS, Zheng YT, Li HT, et al., 2018. Control design for output tracking of delayed Boolean control networks. *J Comput Appl Math*, 327:188-195. https://doi.org/10.1016/j.cam.2017.06.016

Lu JQ, Zhong J, Huang C, et al., 2016. On pinning controllability of Boolean control networks. *IEEE Trans Autom Contr*, 61(6):1658-1663. https://doi.org/10.1109/TAC.2015.2478123

Lu JQ, Li HT, Liu Y, et al., 2017. Survey on semi-tensor product method with its applications in logical networks and other finite-valued systems. *IET Contr Theory Appl*, 11(13):2040-2047. https://doi.org/10.1049/iet-cta.2016.1659

Lu JQ, Li ML, Liu Y, et al., 2018a. Nonsingularity of Grain-like cascade FSRs via semi-tensor product. *Sci China Inform Sci*, 61:010204. https://doi.org/10.1007/s11432-017-9269-6

Lu JQ, Sun LJ, Liu Y, et al., 2018b. Stabilization of Boolean control networks under aperiodic sampled-data control. *SIAM J Contr Optim*, 56(6):4385-4404. https://doi.org/10.1137/18M1169308

Lu JQ, Li ML, Huang TW, et al., 2018c. The transformation between the Galois NLFSRs and the Fibonacci NLFSRs via semi-tensor product of matrices. *Automatica*, 96:393-397. https://doi.org/10.1016/j.automatica.2018.07.011

Ostrowski M, Paulevé L, Schaub T, et al., 2016. Boolean network identification from perturbation time series data combining dynamics abstraction and logic programming. *Biosystems*, 149:139-153. https://doi.org/10.1016/j.biosystems.2016.07.009

Shmulevich I, Dougherty ER, Zhang W, 2002a. From Boolean to probabilistic Boolean networks as models of genetic regulatory networks. *Proc IEEE*, 90(11):1778-1792. https://doi.org/10.1109/JPROC.2002.804686

Shmulevich I, Dougherty ER, Zhang W, 2002b. Control of stationary behavior in probabilistic Boolean networks by means of structural intervention. *J Biol Syst*, 10(4):431-445. https://doi.org/10.1142/S0218339002000706

Shmulevich I, Dougherty ER, Zhang W, 2002c. Gene perturbation and intervention in probabilistic Boolean networks. *Bioinformatics*, 18(10):1319-1331. https://doi.org/10.1093/bioinformatics/18.10.1319

Wang B, Feng JE, 2019. On detectability of probabilistic Boolean networks. *Inform Sci*, 483:383-395. https://doi.org/10.1016/j.ins.2019.01.055

Wang B, Feng JE, Meng M, 2019. Model matching of switched asynchronous sequential machines via matrix approach. *Int J Contr*, 92(10):2430-2440. https://doi.org/10.1080/00207179.2018.1441552

Xiao YF, Dougherty ER, 2007. The impact of function perturbations in Boolean networks. *Bioinformatics*, 23(10):1265-1273. https://doi.org/10.1093/bioinformatics/btm093

Xu XJ, Li HT, Li YL, et al., 2018. Output tracking control of Boolean control networks with impulsive effects. *Math Methods Appl Sci*, 41(4):1554-1564. https://doi.org/10.1002/mma.4685

Yu YY, Feng JE, Pan JF, et al., 2019a. Block decoupling of Boolean control networks. *IEEE Trans Autom Contr*, 64(8):3129-3140. https://doi.org/10.1109/TAC.2018.2880411

Yu YY, Wang B, Feng JE, 2019b. Input observability of Boolean control networks. *Neurocomputing*, 333:22-28. https://doi.org/10.1016/j.neucom.2018.12.014

Zhang LQ, Feng JE, Feng XH, et al., 2014. Further results on disturbance decoupling of mix-valued logical networks. *IEEE Trans Autom Contr*, 59(6):1630-1634. https://doi.org/10.1109/TAC.2013.2292733

Zhao Y, Kim J, Filippone M, 2013. Aggregation algorithm towards large-scale Boolean network analysis. *IEEE Trans Autom Contr*, 58(8):1976-1985. https://doi.org/10.1109/TAC.2013.2251819

Zhao Y, Ghosh BK, Cheng DZ, 2016. Control of large-scale Boolean networks via network aggregation. *IEEE Trans Neur Netw Learn Syst*, 27(7):1527-1536. https://doi.org/10.1109/TNNLS.2015.2442593

Zhong J, Li BW, Liu Y, et al., 2020. Output feedback stabilizer design of Boolean networks based on network structure. *Front Inform Technol Electron Eng*, 21(2):247-259. https://doi.org/10.1631/FITEE.1900229

Zhu QX, Liu Y, Lu JQ, et al., 2018. On the optimal control of Boolean control networks. *SIAM J Contr Optim*, 56(2):1321-1341. https://doi.org/10.1137/16M1070281