第 38 卷第 11 期
2021 年 11 月

控 制 理 论 与 应 用
Control Theory & Applications

Vol. 38 No. 11
Nov. 2021

状态翻转控制下布尔控制网络的可镇定性和Q学习算法

刘 洋[1,2,3†], 刘泽娇[1,3], 卢剑权[4]

(1. 浙江师范大学 数学与计算机科学学院, 浙江 金华 321004; 2. 浙江师范大学 数理医学院, 浙江 金华 321004;

3. 金华市智能制造研究院, 浙江 金华 321032; 4. 东南大学 数学学院, 江苏 南京 210096)

摘要: 在给定一个子集的条件下, 本文研究了在状态翻转控制下布尔控制网络的全局镇定问题. 对于节点集的给定子集, 状态翻转控制可以将某些节点的值从1 (或0)变成0 (或1). 将翻转控制作为控制之一, 本文研究了状态翻转控制下的布尔控制网络. 将控制输入和状态翻转控制结合, 提出了联合控制对和状态翻转转移矩阵的概念. 接着给出了状态翻转控制下布尔控制网络全局稳定的充要条件. 镇定核是最小基数的翻转集合, 本文提出了一种寻找镇定核的算法. 利用可达集的概念, 给出了一种判断全局镇定和寻找联合控制对序列的方法. 此外, 如果系统是一个大型网络, 则可以利用一种名为Q学习算法的无模型强化学习方法寻找联合控制对序列. 最后给出了一个数值例子来说明本文的理论结果.

关键词: 布尔控制网络; 半张量积; 状态翻转控制; 全局镇定性; Q学习算法

引用格式: 刘洋, 刘泽娇, 卢剑权. 状态翻转控制下布尔控制网络的可镇定性和Q学习算法. 控制理论与应用, 2021, 38(11): 1743 – 1753

DOI: 10.7641/CTA.2021.10795

State-flipped control and Q-learning algorithm for the stabilization of Boolean control networks

LIU Yang[1,2,3†], LIU Ze-jiao[1,3], LU Jian-quan[4]

(1. College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua Zhejiang 321004, China;
2. College of Mathematical Medicine, Zhejiang Normal University, Jinhua Zhejiang 321004, China;
3. Jinhua Intelligent Manufacturing Research Institute, Jinhua Zhejiang 321032, China;
4. School of Mathematics, Southeast University, Nanjing Jiangsu 210096, China)

Abstract: In this paper, the global stabilization of Boolean control networks under state-flipped control with respect to a given subset is addressed. For a given subset of the set of the nodes, the state-flipped control can change the values of some nodes from 1 or 0 to 0 or 1. Considering the flips as controls, Boolean networks under state-flipped control are studied. Combining control inputs with state-flipped controls, the concepts of joint control pair and the state-flipped-transition matrix are proposed. A necessary and sufficient condition is provided to check whether a Boolean control network under state-flipped control can be globally stabilized. An algorithm is developed to find the stabilizing kernel, which is the flip set with the minimal cardinal number. By using the reachable set, another method is provided for global stabilization and joint control pair sequences. Besides, if the system is a large scale network, a model-free reinforcement learning method called Q-learning algorithm, is used for the joint control pair sequences. A numerical example is given to illustrate the theoretical results.

Key words: Boolean control networks; semi-tensor product; state-flippped control; global stabilization; Q-learning algorithm

Citation: LIU Yang, LIU Zejiao, LU Jianquan. State-flipped control and Q-learning algorithm for the stabilization of Boolean control networks. *Control Theory & Applications*, 2021, 38(11): 1743 – 1753

1 Introduction

By modeling the gene as a binary (on-off) device, Kauffman in 1969 firstly proposed Boolean networks (BNs) for investigating different metabolic behaviors of genes [1] that enable to capture the properties of large-scale complex networks [2]. In order to make the BNs applicable to more types of biological networks, control inputs are added into BNs, and they are extended to Boolean control networks (BCNs). Cheng et al. proposed the semi-tensor product (STP) [3], which is

Received 25 August 2021; accepted 22 November 2021.
†Corresponding author. E-mail: liuyang@zjnu.edu.cn; Tel.: +86 15067991513
Recommended by Associate Editor: HONG Yi-guang.
Supported by the National Natural Science Foundation of China (62173308, 61973078) and the Natural Science Foundation of Zhejiang Province of China (LR20F030001, LD19A010001).

a powerful tool in studying BNs (BCNs). Under the framework of STP, BCNs are expressed as finite state discrete time nonlinear dynamic systems with matrix algebra form. On the basis of the matrix algebra form, many properties of BCNs have been investigated, such as stabilization, controllability, observability, fault detection, disturbance decoupling and so on [4–12].

Stabilization problem is an important issue in control theory. In a BCN, stabilization is defined as finding feasible control sequences for any initial state to reach a target fixed point after finite time steps. There are many interesting results in the stabilization problems of BCNs. For example, Li et al. investigated state feedback stabilization for BCNs. Based on the concept of invariant subsets in [13], several necessary and sufficient conditions for set stabilization of BCNs have been presented. By using a minimal number of controllers, Lu et al. studied the pinning stabilization of BCNs in [14].

State-flipped control is a newly control mechanism with little intervention on the system [15–16]. It works by changing the value of some nodes in BCNs from 1 to 0, or from 0 to 1, which simulates turning on or off genes in biological systems. Thanks to its ease of operation, many researchers have adopted the state-flipped control. For instance, Rafimanzelat et al. [16] studied the attractor stabilizability of BNs by flipping some nodes of the state in attractors once, after the networks have passed their transient period in the attractors. Rafimanzelat et al. [15] investigated the attractor controllability of BNs by flipping a subset of nodes in the states of several attractors as well. Chen et al. provided the criteria of controllability and stabilization of BCNs by flipping a subset of nodes in some initial states, rather than flip the nodes of the attractors after the system has passed the transient period. More recently, Zhang et al. have applied the flipping mechanism to the stabilization and set stabilization of switched BCNs, which considers flipping a subset of nodes of initial state once [17]. In addition, the weak stabilization of BNs with flip sequences is investigated in [18]. Up to now, many researchers have implemented state-flipped control into stabilization and controllability of BNs (BCNs).

Reinforcement learning (RL) is one of the methodologies of machine learning, which is used to describe and solve the problems that agents use learning strategies to maximize returns or achieve specific goals in the process of interacting with the environment. As a breakthrough in reinforcement learning algorithms, Q-learning (QL) algorithm was first proposed by Watkins in 1992 [19]. QL algorithm is a model-free RL algorithm, which can be used in the tracking control of autonomous surface vehicles [20], smart grid devices [21], and intelligent intersection traffic signal control [22] and so on. QL can be used to judge some properties of gene regulatory networks, which can reduce the computational complexity to a certain extent compared with the traditional STP method. An important concept in QL is the Q table, which is a mapping table between states-actions and estimated future rewards. Under some conditions, the Q table will converge to a Q^* table where we can read the optimal policy from. QL algorithm was applied to probabilistic Boolean control networks (PBCNs), which shows the advantages of the algorithm in the case of model-free [23]. It investigated the feedback stabilization problem of PBCNs, and compared the STP method with the value iteration method. Acernese et al. [24] also developed a QL algorithm about self-triggered control co-design for stabilization of PBCNs.

The existing works in the state-flipped control mainly consider flipping a subset of nodes just once, no matter for the initial states or the states in attractors. In this paper, we consider the joint control pair, which consists of a state-flipped control and a control input. When studying the global stabilization of BCN under state-flipped control, we first give a set of nodes that can be flipped, and the actual state-flipped control depended on the subset of the given set. There exist many different joint control pairs, hence we propose the concept of joint control pair sequences. In our joint control pair sequences, sometimes the obtained state-flipped control is with respect to an empty set, which means that we do not need to add state-flipped control for the state and it is enough to just take a control input, i.e., a normal case in BCNs. The contributions of this paper are summarized as follows:

- We propose the concept of joint control pair which consists of a state-flipped control and a control input, and apply it into BCNs.
- The global stabilization of BCNs under state-flipped control is studied, and several necessary and sufficient conditions are presented.
- For stabilizing a BCN under state-flipped control, a QL algorithm is designed to find the corresponding joint control pair sequence for every initial state.

2 Preliminaries

2.1 Notations

$\mathcal{D} = \{0, 1\}$ and $\mathbb{R}_{p\times q}$ denotes the set of $p \times q$-dim real matrices. $[m : n] := \{m, m+1, \cdots, n\}$, where $m, n \in \mathbb{N}_+$ with $m \leqslant n$. For a matrix $A = (a_{ij}) \in \mathbb{R}_{p\times q}$ (a_{ij} is the (i, j)-th entry of A) and $c \in \mathbb{R}$, $A > c$ means $a_{ij} > c, \forall i \in [1 : p], j \in [1 : q]$. $\varDelta_n := \{\delta_n^i | 1 \leqslant i \leqslant n\}$, where δ_n^i denotes the i-th column of identity matrix I_n. $\delta_{2^n}^{i_1,i_2,\cdots,i_k}$ is a Boolean vector which equals $\sum\limits_{j=1}^{k} \delta_{2^n}^{i_j}$. $D(\delta_{2^n}^{i_1,i_2,\cdots,i_k}) := \{\delta_{2^n}^{i_1}, \delta_{2^n}^{i_2}, \cdots, \delta_{2^n}^{i_k}\}$ denotes a decomposition of vector $\delta_{2^n}^{i_1,i_2,\cdots,i_k}$. $k\delta_{2^n}^{i,j}$ is a column vector with its i-th and j-th entries being

k, and the remainders are 0 s. A matrix with the form $L = [\delta_m^{i_1}\ \delta_m^{i_2}\ \cdots\ \delta_m^{i_n}]$ is called a logical matrix, simplified as $L = \delta_m[i_1\ i_2\ \cdots\ i_n]$. The collection of logical matrices with dimension $m \times n$ is denoted by $\mathcal{L}_{m\times n}$. $|B|$ denotes the cardinal number of set B, and $\mathcal{P}_B := \{A|A \subseteq B\}$ denotes the power set of B. $\otimes$ denotes the Kronecker product.

2.2 BCN and its algebraic representation

Consider a BCN as follows:

$$X^i(t+1) = f_i(X(t), U(t)), \tag{1}$$

where $X^i(t) \in \mathcal{D}, i \in [1:n]$ denotes the i-th node of state $X(t)$, and $f_i : \mathcal{D}^{m+n} \mapsto \mathcal{D}, i \in [1:n]$ is a logical function. $X(t) = (X^1(t), \cdots, X^n(t)) \in \mathcal{D}^n$, $U(t) = (U_1(t), \cdots, U_m(t)) \in \mathcal{D}^m$ represent the state and the control input of the network (1), respectively. $[1:n]$ is the superscript set of the set of nodes of BCN (1). In this brief, nodes are simply denoted by their superscripts. In addition, we default that Boolean variable 1 in $\mathcal{D}$ is equivalent to the canonical vector δ_2^1 and Boolean variable 0 in $\mathcal{D}$ is equivalent to the canonical vector δ_2^2.

Definition 1 [3] For matrices $A \in \mathbb{R}_{p\times q}$ and $B \in \mathbb{R}_{s\times t}$, the semi-tensor product (STP) with symbol "$\ltimes$" is defined as

$$A \ltimes B = (A \otimes I_{\alpha/q})(B \otimes I_{\alpha/s}), \tag{2}$$

where $\alpha = l\ \mathrm{cm}(q, s)$ is the least common multiple of q and s.

For convenience, the symbol "$\ltimes$" can be omitted when there is no ambiguity. Using the semi-tensor product, state $X = (X^1, X^2, \cdots, X^n) \in \mathcal{D}^n$ can be transformed to its equivalent algebraic representation $x := \ltimes_{i=1}^n x^i \in \Delta_{2^n}$, in which $X^i \in \mathcal{D} \sim x^i \in \Delta_2$, $i \in [1:n]$. Similarly, $U = (U_1, \cdots, U_m) \in \mathcal{D}^m \sim u := \ltimes_{i=1}^m u_i \in \Delta_{2^m}$.

Next, based on STP and its properties in [3], we can convert system (1) into its algebraic form as

$$x(t+1) = Gu(t)x(t), \tag{3}$$

where $G \in \mathcal{L}_{2^n\times 2^{m+n}}$ is the state transition matrix of BCN (1). Define $G(u(t)) := Gu(t) \in \mathcal{L}_{2^n\times 2^n}$ as the control-depending network transition matrix of BCN (1). If $u(t) = \delta_{2^m}^q$, then we use G_q to represent $G\delta_{2^m}^q$, i.e., $G_q = G\delta_{2^m}^q$. In addition, let $M := \sum\limits_{q=1}^{2^m} G_q \in \mathbb{R}_{2^n\times 2^n}$. Then, the reachability between any two states can be obtained by the matrix M and its power [5].

2.3 State-flipped control

This subsection introduces the state-flipped control and its algebraic representation. Before introducing the flip function, we need to choose a flip set, which is a subset of the nodes' set of a BCN. It can be specified in random, or it can be a collection of genes that we are able to control in the practical cases.

Definition 2 Let $A := \{a_1, a_2, \cdots, a_r\} \subseteq [1:n]$. The flip function with respect to A is defined as

$$\begin{aligned}\eta_A^{\neg}(X) = (&X^1, \cdots, \neg X^{a_1}, \cdots, \neg X^{a_2}, \cdots,\\ &\neg X^{a_r}, \cdots, X^n).\end{aligned} \tag{4}$$

The flip function can transform one state to another state $X_A^{\neg} = \eta_A^{\neg}(X)$. According to Definition 2, we can obtain that X and $X_A^{\neg}$ can be converted to each other by flipping A, i.e., $X \overset{\eta_A^{\neg}}{\leftrightarrow} X_A^{\neg}$. Here, we call (4) a state-flipped control, and A is a flip set. Next, based on the equivalence of the logical representation of the state and its vector representation, we define the matrix form of the flip function.

Definition 3 Let $A := \{a_1, a_2, \cdots, a_r\} \subseteq [1:n]$. The algebraic matrix form of $\eta_A^{\neg}$ denoted by $\mathcal{H}_A$ is called a flip matrix, which satisfies:

$$\begin{aligned}&\mathrm{Col}_j(\mathcal{H}_A) = \delta_{2^n}^i,\ j \in [1:2^n],\\ &\text{if } x = \delta_{2^n}^j \overset{\eta_A^{\neg}}{\to} x_A^{\neg} = \delta_{2^n}^i.\end{aligned} \tag{5}$$

From the definition of $\mathcal{H}_A$, we can derive that $\mathcal{H}_A$ is a symmetric matrix, and it includes all cases that every state in Δ_{2^n} is flipped with respect to A. Note that $\mathcal{H}_A$ is a $2^n \times 2^n$-dim logical matrix, and equation (5) can be expressed as $x_A^{\neg} = \mathcal{H}_A x$. In this paper, we call the transition from state x to state x_A as a flip transition. After introducing flip matrix with respect to a set A, we can consider the cases that multiple sets of nodes can be chosen to be flipped. In the following, B represents a set of nodes that can be flipped, and we can choose a few of these nodes to flip.

Definition 4 Let $B := \{b_1, b_2, \cdots, b_s\} \subseteq [1:n]$. The combinatorial flip matrix with respect to B is defined as

$$(\mathcal{C}_B)_{ij} = \begin{cases}1, \text{ if } \exists A \in \mathcal{P}_B \text{ such that } x = \delta_{2^n}^j \overset{\eta_A^{\neg}}{\to} x_A^{\neg} = \delta_{2^n}^i,\\ 0, \text{ otherwise.}\end{cases}$$

Combinatorial flip matrix $\mathcal{C}_B$ contains flip cases that each subset of B is flipped. $(\mathcal{C}_B)_{ij} = 1$ means that there exists one subset $A \subseteq B$ such that $(\mathcal{H}_A)_{ij} = 1$. For any initial state $x_0 = \delta_{2^n}^j \in \Delta_{2^n}$, when the flip sets $A_1, A_2 \subseteq B$ are different, we have $\delta_{2^n}^{i_1} = x_{A_1}^{\neg} \neq x_{A_2}^{\neg} = \delta_{2^n}^{i_2}$. Therefore, the j-th column of $\mathcal{C}_B$ doesn't have an entry greater than 1. It illustrates that $\mathcal{C}_B$ is a Boolean matrix. In order to represent the above in mathematical notations, we can derive:

$$\mathcal{C}_B = \sum_{A\in\mathcal{P}_B} \mathcal{H}_A \in \mathcal{B}_{2^n\times 2^n}. \tag{6}$$

In equation (6), $\mathcal{C}_B$ is the combination of all possible subsets of $B = \{b_1, b_2, \cdots, b_s\} \subseteq [1:n]$ whose corresponding nodes are chosen to be flipped. Based on Definition 4, we can find that $D(\mathrm{Col}_j(\mathcal{C}_B))$, $j \in [1:2^n]$ contains all the states that can be reached from $x = \delta_{2^n}^j$ by flipping every subset of B. In this paper, we first consider that all states in BCN (3) are flipped with respect to all subsets of B. The new system that all states take one flip transition with respect to one subset of B and one state transition with control input is called BCN (3) under B state-flipped control.

Here, we present a brief explanation of two sets A and B. For a given state, A is the actual flip set, and every element in A corresponding to the nodes of the given state has to be flipped. However, B is a combinatorial flip set, and we select a subset of B denoted by A to flip. Next, we give a numerical example of a BCN (3) to illustrate the flip matrix and the combinatorial flip matrix.

Example 1 Consider a BCN with three nodes, i.e. $n = 3$. If a subset $B \subseteq [1:n]$ is given by $B = \{2,3\}$, we can find that all possible flip sets are subsets of B, denoted by $A_1 = \varnothing, A_2 = \{2\}, A_3 = \{3\}, A_4 = \{2,3\}$. Then, flip matrices with flip sets $A_i, i = [1:4]$ can be calculated as

$$\mathcal{H}_{A_1} = \mathcal{H}_{\varnothing} = I_8 = \delta_8[1\ 2\ 3\ 4\ 5\ 6\ 7\ 8],$$
$$\mathcal{H}_{A_2} = \mathcal{H}_{\{2\}} = \delta_8[3\ 4\ 1\ 2\ 7\ 8\ 5\ 6],$$
$$\mathcal{H}_{A_3} = \mathcal{H}_{\{3\}} = \delta_8[2\ 1\ 4\ 3\ 6\ 5\ 8\ 7],$$
$$\mathcal{H}_{A_4} = \mathcal{H}_{\{2,3\}} = \delta_8[4\ 3\ 2\ 1\ 8\ 7\ 6\ 5].$$

Therefore, we can obtain that the combinatorial flip matrix $\mathcal{C}_B$ is

$$\mathcal{C}_B = \sum_{A\in\mathcal{P}_B}\mathcal{H}_A = \sum_{i=1}^{4}\mathcal{H}_{A_i} = [\delta_8^{1,2,3,4}\ \delta_8^{1,2,3,4}\ \delta_8^{1,2,3,4}\ \delta_8^{1,2,3,4}\ \delta_8^{5,6,7,8}\ \delta_8^{5,6,7,8}\ \delta_8^{5,6,7,8}\ \delta_8^{5,6,7,8}].$$

3 Main results

This section focuses on the stabilization of BCN (3) under B state-flipped control. Several criteria are proposed to judge the stabilization. Note that the global state transition space may be changed under state-flipped control defined in Section 2. Hence, we present a new type of the state transition matrix, which is called state-flipped-transition matrix.

Definition 5 Given a matrix $A = (a_{ij}) \in \mathbb{R}_{m\times n}$, define $\operatorname{sgn}(A) := (\operatorname{sgn}(a_{ij}))$ with

$$\operatorname{sgn}(a_{ij}) = \begin{cases} 1, & a_{ij} > 0, \\ 0, & a_{ij} = 0, \\ -1, & a_{ij} < 0. \end{cases}$$

Definition 6 Given a subset $B \subseteq [1:n]$. The matrix $\widetilde{G} \in \mathbb{R}_{2^n\times 2^n}$ is called the state-flipped-transition matrix of BCN (3) under B state-flipped control, if

$$\widetilde{G} = M\mathcal{C}_B. \tag{7}$$

According to $\widetilde{G}$, the new system transformed from BCN (3) under B state-flipped control is

$$z(t+1) = \operatorname{sgn}(\widetilde{G}z(t)). \tag{8}$$

Therefore, $z(t)$ is a Boolean vector with several entries equal to 1 and other entries equal to 0. Set $z(0) = x(0) = x_0$. The state transition between two states is called a state-flipped transition in BCN (3) under B state-flipped control with the combined action of a state-flipped control and a control input. Here, we use notation $(\eta_A^{-}, \delta_{2^m}^i)$ which is called the joint control pair to represent the combined action in one state-flipped transition step of BCN (3). $D(z(t+1))$ represents the set of all states steered from $D(z(t))$ after one state-flipped transition step.

For the state-flipped transitions from $\delta_{2^n}^j$ to $\delta_{2^n}^i$, we denote the joint control pair sequence composed of some joint control pairs by

$$\Lambda_{\{\delta_{2^n}^j,\delta_{2^n}^i\}} := \{(\eta_{A_0}^{-}, u_0), (\eta_{A_1}^{-}, u_1), \cdots, (\eta_{A_{k-1}}^{-}, u_{k-1})\},$$

which is also denoted by $\boldsymbol{\Lambda}_k$, where $A_j \subseteq B$ is a flip set, and $u_j \in \Delta_{2^m}$ is a control input, $j \in [0:k-1]$. Using the information in $\Lambda_{\{\delta_{2^n}^j,\delta_{2^n}^i\}}$, we can obtain a state-flipped transition walk as

$$P = \{x_0 = \delta_{2^n}^j \xrightarrow{(\eta_{A_0}^{-},u_0)} x_1 = \delta_{2^n}^{p_1} \xrightarrow{(\eta_{A_1}^{-},u_1)} x_2 = \delta_{2^n}^{p_2} \xrightarrow{(\eta_{A_2}^{-},u_2)} \cdots \xrightarrow{(\eta_{A_{k-1}}^{-},u_{k-1})} x_k = \delta_{2^n}^i\}.$$

Remark 1 $\delta_{2^n}^d \in \Delta_{2^n}$ is said to be a fixed point if there exists a state-flipped transition from $\delta_{2^n}^d$ to itself. $\{\delta_{2^n}^{a_1}, \delta_{2^n}^{a_2}, \cdots, \delta_{2^n}^{a_q}\} \subseteq \Delta_{2^n}$ is a cycle with length q, if it satisfies that there always exists at least one state-flipped transition from $\delta_{2^n}^{a_i}$ to $\delta_{2^n}^{a_{i+1}}, i \in [1:q-1]$, and one state-flipped transition from $\delta_{2^n}^{a_q}$ to $\delta_{2^n}^{a_1}$. In BCN (3) without flipping, if there exists a control input such that there is a state $x(t) = \delta_{2^n}^j$ can be steered to $x(t+1) = \delta_{2^n}^i$, then we say that the in-degree of state $\delta_{2^n}^i$ is greater than 0. In addition, if the in-degree of a state is 0, then there does not exist any state-flipped transition to this state. Therefore, in the further consideration of the stabilization problem, we assume that the in-degree of a given state is greater than 0.

Definition 7 Given a subset $B \subseteq [1:n]$. For an initial state $x_0 \in \Delta_{2^n}$, let $x(k; \boldsymbol{u}_k, x_0)$ be the state of BCN (3) at time k, where $\boldsymbol{u}_k = \{u_0, u_1, \cdots, u_{k-1}\}$ is a control input sequence. Let $x(k; \boldsymbol{\Lambda}_k, x_0)$ be the state of BCN (3) under B state-flipped control at time k, where $\boldsymbol{\Lambda}_k$ is the joint control pair sequence with k joint control pairs.

Theorem 1 For any joint control pair sequence $\boldsymbol{\Lambda}_t$, assume that $x_0 \in \Delta_{2^n}$ is an initial state, then the state reached from x_0 after some state-flipped transitions with joint control pairs is always in $D(z(t))$, i.e., $x(t; \boldsymbol{\Lambda}_t, x_0) \in D(z(t))$.

Proof We give the proof by mathematical induction. For the case $t = 1$, based on Definition 7, we can obtain that $x(1; \boldsymbol{\Lambda}_1, x_0)$ is a state of BCN from x_0 after one state-flipped transition step. Then, $x(1; \boldsymbol{\Lambda}_1, x_0) \in D(\text{sgn}(M\mathcal{C}_B x_0))$. Since $D(\text{sgn}(M\mathcal{C}_B x_0)) = D(\text{sgn}(\widetilde{G}x_0)) = D(\text{sgn}(\widetilde{G}z(0))) = D(z(1))$, we have $x(1; \boldsymbol{\Lambda}_1, x_0) \in D(z(1))$. Next, suppose that $x(t; \boldsymbol{\Lambda}_t, x_0) \in D(z(t))$ holds for the case $t = k$. When $t = k + 1$, $x(k + 1; \boldsymbol{\Lambda}_{k+1}, x_0) = x(1; \boldsymbol{\Lambda}_1, x_k)$, where $x_k = x(k; \boldsymbol{\Lambda}_k, x_0) \in D(z(k))$. Let $D(z(k)) = \{\delta_{2^n}^{i_1}, \delta_{2^n}^{i_2}, \cdots, \delta_{2^n}^{i_p}\}$ and take $x_k = \delta_{2^n}^{i_k}$. Then, we have that $x(1; \boldsymbol{\Lambda}_1, \delta_{2^n}^{i_k}) \in D(\text{sgn}(\widetilde{G}\delta_{2^n}^{i_k})) \subseteq D(z(k+1))$. Hence, $x(k + 1; \boldsymbol{\Lambda}_{k+1}, x_0) \in D(z(k + 1))$. □

In Theorem 1, an approach for calculating $x(t; \boldsymbol{\Lambda}_t, x_0)$ is given by calculating $z(t)$ first in BCN (3) under B state-flipped control. Further, we provide the significance of each component of $\widetilde{G}$. Let $w(k; \delta_{2^n}^j, \delta_{2^n}^i)$ be the number of the ways to steer BCN (3) under B state-flipped control from the initial state $x_0 = \delta_{2^n}^j$ to the destination state $x_k = \delta_{2^n}^i$ after k state-flipped transition steps. Next, we derive a theorem to obtain the number of the ways $w(k; \delta_{2^n}^j, \delta_{2^n}^i)$ by using state-flipped-transition matrix $\widetilde{G}$.

Theorem 2 Let $B := \{b_1, b_2, \cdots, b_s\} \subseteq [1 : n]$ and $\widetilde{G} = M\mathcal{C}_B$ be the state-flipped-transition matrix defined as (7). Then, there is $[(\widetilde{G})^k]_{ij} = w(k; \delta_{2^n}^j, \delta_{2^n}^i)$.

Proof According to the definition of state-flipped transition matrix $\widetilde{G}$, we have

$$
\begin{aligned}
(\widetilde{G})_{ij} = (M\mathcal{C}_B)_{ij} = \\
&\text{Row}_i(M)\text{Col}_j(\mathcal{C}_B) = \\
&\text{Row}_i(\sum_{q=1}^{2^m} G_q)\text{Col}_j(\sum_{A\in\mathcal{P}_B} \mathcal{H}_A) = \\
&\sum_{q=1}^{2^m} \text{Row}_i(G_q) \sum_{A\in\mathcal{P}_B} \text{Col}_j(\mathcal{H}_A) = \\
&\sum_{q=1}^{2^m} \sum_{A\in\mathcal{P}_B} \text{Row}_i(G\delta_{2^m}^q)(\mathcal{H}_A\delta_{2^n}^j) = \\
&\sum_{q=1}^{2^m} \sum_{A\in\mathcal{P}_B} (\delta_{2^n}^i)^\top (G\delta_{2^m}^q)(\mathcal{H}_A\delta_{2^n}^j).
\end{aligned}
$$

If $x(1; \boldsymbol{\Lambda}_1, \delta_{2^n}^j) = \delta_{2^n}^i$ with $\boldsymbol{\Lambda}_1 = (\eta_A^{\neg}, \delta_{2^m}^q)$, then we have $(\delta_{2^n}^i)^\top (G\delta_{2^m}^q)(\mathcal{H}_A\delta_{2^n}^j) = 1$. Therefore, $(\widetilde{G})_{ij} = \sum_{q=1}^{2^m} \sum_{A\in\mathcal{P}_B} 1$ with the condition $x(1; \boldsymbol{\Lambda}_1, \delta_{2^n}^j) = \delta_{2^n}^i$. For the case $t = k$, we have $[(\widetilde{G})^k]_{ip} = w(k; \delta_{2^n}^p, \delta_{2^n}^i)$. Next, for the case $t = k + 1$, we can conclude

$$
\begin{aligned}
(\widetilde{G})_{ij}^{k+1} = &\sum_{p=1}^{2^n} [(\widetilde{G})^k]_{ip} \widetilde{G}_{pj} = \\
&\sum_{p=1}^{2^n} w(k; \delta_{2^n}^p, \delta_{2^n}^i) w(1; \delta_{2^n}^j, \delta_{2^n}^p) = \\
&w(k + 1; \delta_{2^n}^j, \delta_{2^n}^i).
\end{aligned}
$$

Thus, $(\widetilde{G})_{ij}^k$ can be used to calculate the number of ways from state $\delta_{2^n}^j$ to state $\delta_{2^n}^i$ after k state-flipped transition steps. □

Based on Theorem 2, we can obtain the reachability between any two states in the BCN (3) under B state-flipped control based on the calculation of the matrix $\widetilde{G}$ and its powers. $(\widetilde{G})_{ij} > 0$ implies that there exists at least a set $A \in \mathcal{P}_B$ and a control-depending network transition matrix G_q, such that $\delta_{2^n}^i = G_q\mathcal{H}_A\delta_{2^n}^j$.

Now, we give an example to illustrate the validity of Theorem 2.

Example 2 Given $B = \{2, 3\}$ as is mentioned in Example 1. Consider a BCN with three nodes and one control input, i.e. $n = 3, m = 1$ in [14]. Its algebraic representation is

$$x(t + 1) = Gu(t)x(t), \tag{9}$$

where $G = \delta_8[2\ 1\ 1\ 5\ 5\ 2\ 1\ 7\ 1\ 2\ 1\ 5\ 5\ 1\ 1\ 7]$. Then, $M = \sum_{q=1}^{2} G_q = [\delta_8^{1,2}\ \delta_8^{1,2}\ 2\delta_8^1\ 2\delta_8^5\ 2\delta_8^5\ \delta_8^{1,2}\ 2\delta_8^1\ 2\delta_8^7]$. Recall $\mathcal{C}_B$ in Example 1, we have

$$
\begin{aligned}
\widetilde{G} = M\mathcal{C}_B = [&4\delta_8^1 + 2\delta_8^{2,5}\ 4\delta_8^1 + 2\delta_8^{2,5}\ 4\delta_8^1 + \\
&2\delta_8^{2,5}\ 4\delta_8^1 + 2\delta_8^{2,5}\ 3\delta_8^1 + \delta_8^2 + 2\delta_8^{5,7}\ 3\delta_8^1 + \\
&\delta_8^2 + 2\delta_8^{5,7}\ 3\delta_8^1 + \delta_8^2 + 2\delta_8^{5,7}\ 3\delta_8^1 + \delta_8^2 + 2\delta_8^{5,7}].
\end{aligned}
$$

To show the validity of Theorem 2, we can find that $\widetilde{G}_{7,5} = 2$, which implies that in the BCN (9) under $\{2, 3\}$ state-flipped control, there are 2 ways for δ_8^5 to be steered to δ_8^7 after one state-flipped transition step. In fact, we can only find $(M\mathcal{H}_{\{2,3\}})_{7,5} = 2$. Since $\mathcal{H}_{\{2,3\}}\delta_8^5 = \delta_8^8$, and $G_1\delta_8^8 = \delta_8^7, G_2\delta_8^8 = \delta_8^7$, we get that the joint control pair from δ_8^5 to δ_8^7 after one state-flipped transition step is $(\eta_{\{2,3\}}^{\neg}, \delta_2^1)$ or $(\eta_{\{2,3\}}^{\neg}, \delta_2^2)$.

Next, after introducing matrix $\widetilde{G}$, we are able to address the global stabilization of BCN (3) under B state-flipped control based on $\widetilde{G}$. We derive the definition of x_d stabilizable for a BCN (3) under B state-flipped control as follows, where the in-degree of the state x_d is greater than 0.

Definition 8 For a given target state $x_d = \delta_{2^n}^d \in \Delta_{2^n}$, BCN (3) under B state-flipped control is said to be globally stabilizable to x_d, if for any $x_0 \in \Delta_{2^n}$, there exists a joint control pair sequence $\boldsymbol{\Lambda}_t$ and a positive integer N, such that for any $t \geqslant N$,

$$x(t; \boldsymbol{\Lambda}_t, x_0) = x_d. \tag{10}$$

If BCN (3) under B state-flipped control can achieve global stabilization, we call the set B stabilizing set.

In Definition 8, we need to find proper $\boldsymbol{\Lambda}_t$ for the global stabilization. The result of Theorem 2 implies that the reachability between two states can be calculated by $\widetilde{G}$. Therefore, we propose Theorem 3 as a prior condition to judge the global stabilizaion of BCN (3) under B state-flipped control. If the global stabilization under state-flipped control cannot be achieved by a subset $B \subseteq [1:n]$ in Theorem 3, then we need to choose another set to replace B.

Theorem 3 For a given subset $B \subseteq [1:n]$ and a state $x_d = \delta_{2^n}^d$, BCN (3) under B state-flipped control is globally stabilizable to x_d if and only if the following two statements hold:

1) $(\widetilde{G})_{dd} > 0$;

2) There exists a positive integer $k \in [1:2^n-1]$, such that $\mathrm{Row}_d((\widetilde{G})^k) > 0$.

Proof [Sufficiency] Owing to Theorem 2, if there exists an integer $k_0 \in [1:2^n-1]$ such that $\mathrm{Row}_d((\widetilde{G})^{k_0}) > 0$, $\delta_{2^n}^d$ can be achieved from $\delta_{2^n}^j, j \in [1:2^n]$ after k_0 state-flipped transition steps. Moreover, $(\widetilde{G})_{dd} > 0$ shows that $x_d = \delta_{2^n}^d$ is a fixed point. Therefore, we can find the positive integer $N = k_0$, and there exists a joint control pair sequence $\boldsymbol{\Lambda}_t$, such that for all $t \geqslant k_0, x(t;\boldsymbol{\Lambda}_t, x_0) = x_d, \forall x_0 \in \Delta_{2^n}$.

(Necessity) Since BCN (3) under B state-flipped control is globally stabilizable to x_d, for all x_0, there exist a joint control pair sequence $\boldsymbol{\Lambda}_t$ and a positive integer N, such that for all $t \geqslant N, x(t;\boldsymbol{\Lambda}_t, x_0) = x_d$. Taking $t = N$ into consideration, for any initial state x_0, we have $x(N;\boldsymbol{\Lambda}_N, x_0) = x_d$. Similarly, it holds that $x(N+1;\boldsymbol{\Lambda}_{N+1}, x_0) = x_d$. Hence, there exists at least one walk with length N from x_0 to x_d in BCN (3) under B state-flipped control, and another walk with length $N+1$ from x_0 to x_d. Therefore, we can conclude that there exists a way in BCN (3) under B state-flipped control from x_d to x_d after one state-flipped transition step. Based on Theorem 2, we have $(\widetilde{G})_{dd} > 0$. Besides, according to Definition 8, there always exists at least one walk from any state $\delta_{2^n}^j, j \in [1:2^n]$ to $\delta_{2^n}^d$. Denote the walk by P_j. Let k_{j_1} represent the length of the walk. It is obvious that $k_{j_1} \geqslant 1$. Then, we need to prove that $k_{j_1} \leqslant 2^n$. If $k_{j_1} \geqslant 2^n$, considering that there are 2^n states in the state space, there must be some cycles in the walk P_j. Remove all cycles in P_j, then we can obtain a simple path from $\delta_{2^n}^j$ to $\delta_{2^n}^d$ with its length less than or equal to 2^n-1. The length of simple path from $\delta_{2^n}^j$ to $\delta_{2^n}^d$ is denoted by k_j. Then, take $k = \max\{k_1, k_2, \cdots, k_{2^n}\} \in [1:2^n-1]$. Due to the arbitrary of $x_0 = \delta_{2^n}^j$, combining $((\widetilde{G})^k)_{dj}$, we can obtain that $\mathrm{Row}_d((\widetilde{G})^k) > 0$. □

If we have verified that BCN (3) under B state-flipped control is globally stabilizable to x_d, then BCN (3) can be also globally stabilizable to x_d under state-flipped control for any superset of B. In order to reduce the control cost, we always expect that the cardinal number of B achieving global stabilization is as small as possible. A stabilizing set B with minimal cardinal number is said to be a stabilizing kernel of BCN (3) under B state-flipped control, and the corresponding minimal "N" in Definition 8 is called stabilizing step. Since the subset B is given in advance, it is used to prejudge whether BCN (3) under B state-flipped control can achieve global stabilization to a given state. However, it is possible that a subset of B might be a better stabilizing set with smaller cardinal number. Hence, it inspires us to find a stabilizing kernel, which is a subset of B, to give the state-flipped control. Algorithm 1 is developed to obtain a stabilizing kernel based on the given B.

Algorithm 1 An algorithm for finding a stabilizing kernel and the corresponding stabilizing step of BCN (3) based on a given set B to achieve global stabilization to $\delta_{2^n}^d$

Input: M, B
Output: B_{γ_i}, k
1: **Initialization**
2: $\gamma = 1$
3: $i = 1$
4: Initialize θ and C_θ^γ
5: **If** $(M\mathcal{C}_{B_{\gamma_i}})_{dd} > 0$, **go to** step 6
6: $k = 1$
7: **If** $\mathrm{Row}_d[(M\mathcal{C}_{B_{\gamma_i}})^k] > 0$,
8: **return** B_{γ_i}, k, **end**
9: **else** $k \leftarrow k+1$
10: **If** $k \leqslant 2^n - 1$, **go to** step 7
11: **else** $i \leftarrow i+1$
12: **If** $i \leqslant C_\theta^\gamma$, **go to** step 5
13: **else go to** step 14
14: **If** output is empty, $\gamma \leftarrow \gamma + 1$
15: **If** $\gamma \leqslant \theta$, **go to** step 3
16: **else end**
17: **else end**
18: **else** $i \leftarrow i+1$
19: **If** $i \leqslant C_\theta^\gamma$, **go to** step 5
20: **else** $\gamma \leftarrow \gamma + 1$
21: **If** $\gamma \leqslant \theta$, **go to** step 3
22: **else end**

Now, we give several explanations of the notations using in Algorithm 1. The cardinal number of given subset B is θ, i.e. $|B| = \theta$. B_{γ_i} is a subset of B with cardinal number being γ. C_θ^γ is a combinatorial number. If B_{γ_i} and k are returned, then B_{γ_i} is a stabilizing kernel and k is its corresponding stabilizing step.

Based on the above analysis, for a traditional BCN (3), if it cannot achieve global stabilization to any state, we can consider adding some state-flipped controls. Given a subset $B \subseteq [1:n]$, the global stabilization with respect to x_d can be checked by Theorem 3 under B state-flipped control. Then, Algorithm 1 presents a method for calculating the stabilizing kernel and the stabilizing step. In practical problems, we not only need to judge whether the network can be globally stabilizable, but also need to find the corresponding joint control pair sequences for each state. Thus, another method about reachable set for global stabilization is provided.

Definition 9 For a given target state $x_d = \delta_{2^n}^d$. In a BCN (3) under B state-flipped control, the k step reachable set of x_d denoted by $E_k(d)$, is defined as:

i) $E_1(d) = \{x_0 | \exists \boldsymbol{\Lambda}_1,$ such that $x(1; \boldsymbol{\Lambda}_1, x_0) = x_d\}$,

ii) $E_{k+1}(d) = \{x_0 \in \overline{E_k(d)}^c \,|\, \exists \boldsymbol{\Lambda}_1$ such that $x(1; \boldsymbol{\Lambda}_1, x_0) \in E_k(d)\}$, where $\overline{E_k(d)} = \bigcup\limits_{i=1}^{k} E_i(d)$, $\overline{E_k(d)}^c = \Delta_{2^n} \backslash \overline{E_k(d)}$.

In the construction of k step reachable set, we can obatin that $E_i(d) \cap E_j(d) = \varnothing$ for any positive integers i and j satisfying $1 \leqslant i \neq j \leqslant 2^n - 1$. Next, we can calculate the k step reachable set of x_d to determine whether BCN (3) under B state-flipped control is globally stabilizable to x_d.

Theorem 4 Given a target state $x_d = \delta_{2^n}^d \in \Delta_{2^n}$. BCN (3) under B state-flipped control is globally stabilizable to x_d, if and only if the following two statements hold:

1) $x_d \in E_1(d)$;

2) There exists a positive integer $N \in [1 : 2^n - 1]$ such that $\bigcup\limits_{k=1}^{N} E_k(d) = \Delta_{2^n}$.

proof If $x_d \in E_1(d)$, then there exists a joint control pair sequence $\boldsymbol{\Lambda}_1$ such that $x(1; \boldsymbol{\Lambda}_1, x_d) = x_d$. It is equal to the first condition in Theorem 3 that $(\widetilde{G})_{dd} > 0$ implies x_d is a fixed point in BCN (3) under B state-flipped control. If there exists a positive integer $N \in [1 : 2^n - 1]$ such that $\bigcup\limits_{k=1}^{N} E_k(d) = \Delta_{2^n}$, then for any initial state $x_0 = \delta_{2^n}^j \in \Delta_{2^n}$, there exists a k steps state-flipped transition to steer x_0 to x_d, $k \in [1 : N]$. Equivalently, it means that we can find N, such that $\mathrm{Row}_d((\widetilde{G})^N) > 0$. According to Theorem 3, we can obtain that BCN (3) under B state-flipped control is globally stabilizable to x_d. From the above analysis, it shows the conditions in Theorem 4 are equal to the conditions in Theorem 3. □

Next, we present Algorithm 2 for calculating the joint control pair sequence we want, which can steer the network from an initial state to the given state. Suppose that we have found the stabilizing kernel is B and $|B| = \theta$. All subsets of B are denoted by $A_1, A_2, \cdots, A_{2^\theta}$. For the state-flipped transitions from $\delta_{2^n}^j$ to $\delta_{2^n}^i$, according to the state-flipped-transition matrix $\widetilde{G}$, we can find a state-flipped transition path $P = \{x_0 = \delta_{2^n}^j \to x_1 = \delta_{2^n}^{p_1} \to x_2 = \delta_{2^n}^{p_2} \to \cdots \to x_k = \delta_{2^n}^i\}$, where x_p is in the $k - p$ step reachable set of x_k. For convenience, suppose that $\delta_{2^n}^{p_0} = \delta_{2^n}^j, \delta_{2^n}^{p_k} = \delta_{2^n}^i$. After we find the path from $\delta_{2^n}^j$ to $\delta_{2^n}^i$, there are several different joint control pair sequences which are feasible. We can calculate $(M\mathcal{H}_{A_{r_t}})_{p_{t+1}, p_t} > 0$ to find the state-flipped control with the flip set A_{r_t} to help steer x_t to x_{t+1}, where $r \in [1 : 2^\theta], t \in [0 : k-1]$. Then, after finding the state-flipped control, one needs to find a control-depending network transition matrix G_{q_t} of BCN (3) to obtain the control input $u_t = \delta_{2^m}^{q_t}$ for x_t. Therefore, a joint control pair $(\eta_{A_{r_t}}^{-}, u_t)$ can be found for each state-flipped transition step from x_t to $x_{t+1}, t \in [0 : k-1]$. Finally, a joint control pair sequence for $\delta_{2^n}^j$ to go to $\delta_{2^n}^i$ is acquired. Since there may have several different joint control pair sequences, note that in this paper we are only interested in the existence of the joint control pair sequences.

Algorithm 2 An algorithm for finding a joint control pair sequence to steer $\delta_{2^n}^j$ to $\delta_{2^n}^i$

Intput: $\delta_{2^n}^j, \delta_{2^n}^i$

Output: $\Lambda_{\{\delta_{2^n}^j, \delta_{2^n}^i\}}$

1: **Initialization**

2: $k = 1$

3: **If** $k \leqslant 2^n - 1$, **do** step 5

4: **else end**

5: **If** $[(\widetilde{G})^k]_{ij} > 0$, then let $k^* = k$, **do** step 7

6: **else** $k \leftarrow k + 1$, **do** step 3

7: Calculate $E_m(i), m \in [1 : k^*]$

8: Find a path $P = \{x_0 = \delta_{2^n}^j \to x_1 = \delta_{2^n}^{p_1} \to x_2 = \delta_{2^n}^{p_2} \to \cdots \to x_{k^*} = \delta_{2^n}^i\}$

9: Calculate $M\mathcal{H}_{A_r}, r \in [1 : 2^\theta]$

10: Find $(M\mathcal{H}_{A_{r_t}})_{p_{t+1}, p_t} > 0$, then the state-flipped control for x_t is $\eta_{A_{r_t}}^{-}$, where $t \in [0 : k^* - 1], r_t \in [1 : 2^\theta]$

11: Find $(G_{q_t}\mathcal{H}_{A_{r_t}})_{p_{t+1}, p_t} = 1$, then the control input for x_t is $u_t = \delta_{2^m}^{q_t}$, where $t \in [0 : k^* - 1], q_t \in [1 : 2^m]$

12: $\Lambda_{\{\delta_{2^n}^j, \delta_{2^n}^i\}} = \{(\eta_{A_{r_0}}^{-}, u_0), (\eta_{A_{r_1}}^{-}, u_1), \cdots, (\eta_{A_{r_{k^*-1}}}^{-}, u_{k^*-1})\}$

13: **end**

If we use BCNs to model large-scale gene regulatory networks, the STP-based approach will have high computational complexity. To this end, we present a QL algorithm, which can be applied in model-free cases and reduces computational complexity, to check whether a BCN (3) under B state-flipped control is globally stabilizable to a given state.

QL algorithm is a type of model-free reinforcement learning algorithm involving Markov decision processes (MDPs), and especially in this paper we only consider the case without any probability [23–24]. As a reinforcement learning method, QL algorithm can achieve goals through interactive learning and training between agents and the environment. Agents can be sensors, drones, power stations in smart grids, gene nodes in biological networks, and so on. The environment representing everything outside the subject can interact with and impact on the agents. Specifically, the agents and the environment constantly interact. The agents select actions, and in turn, the environment responds to those actions and provides new information to the agents. In the process of interaction, the environment generates rewards, namely specific values, which can reflects the quality of the current action. The essence of reinforcement learning is to find a series of actions that maximize the long-term rewards (return) to achieve a given goal. Besides, a policy is a mapping from states to the probability of choosing each possible action. Simply, a policy can be regarded as a choice of actions for a state at each step.

In this paper, we regard a controller as an agent and the unknown system (the BCN) as the environment. A joint control pair is regarded as an action. The reward is set artificially based on a given goal, and the return is the sum of the rewards. After the qualitative introduction, we introduce some specific notations and necessary explanations for the QL algorithm.

In BCN (1) under B state-flipped control, $X_t \in \mathcal{D}^n$ denotes the state at t after t state-flipped transition steps. $X_d \in \mathcal{D}^n$ denotes a given target state. Since both state-flipped control $\eta_{A_t}^{-}$ and control input $U_t \in \mathcal{D}^m$ are adopted, now we recall joint control pair $(\eta_{A_t}^{-}, U_t)$ denoted by $\mathcal{J}_t$ for the sake of convenience.

For the QL algorithm, some basic notations are introduced. $\varLambda^*(X_t, X_d)$ denotes the optimal policy (i.e. a joint control pair sequence achieving the given target state with maximal return) from X_t to X_d. r_t denotes the reward used to calculate the immediate return value received by the agent, after the agent selects an action from the current state and moves to the next state. The reward function is set in advance according to our goals. With target of steering the BCN under B state-flipped control to be globally stabilizable to X_d, we give the setting of the reward r_{t+1} as follows:

$$r_{t+1} = \begin{cases} 100, & X_{t+1} = X_d, \\ 0, & X_{t+1} = X_i \neq X_d, \end{cases} \tag{11}$$

where $X^i \in \mathcal{D}^n$ with $i \in [1:2^n]$.

Based on the above settings, Algorithm 3 is proposed using QL method to check the global stabilization of BCN under B state-flipped control as follows.

In Algorithm 3, $\gamma \in (0,1)$ is the discount factor, which is used to determine the relative ratio of the delayed return to immediate return. α_t denotes the learning rate. When the following two conditions are satisfied, the convergence of Algorithm 3 is guaranteed: i) $\sum\limits_{t=0}^{\infty} \alpha_t = \infty$; ii) $\sum\limits_{t=0}^{\infty} \alpha_t^2 < \infty$, see in [25]. Actually, i) guarantees that the step size is large enough to finally overcome any initial conditions. ii) guarantees that the final step size is small enough to ensure convergence. In this paper, we set $\alpha_t = 1/(t+1)^\omega$ with $0.5 < \omega \leqslant 1$. It can be easily proved that $\alpha_t = 1/(t+1)^\omega$ satisfies conditions i) and ii). The greater the learning rate α_t is, the less the effect of previous training becomes.

Now, we give the origin of Q table in Algorithm 3. We set π to be the policy. $v_\pi(X_t)$ denotes the value function for X_t, which can estimate the long-term discounted return to show the performance of agent at X_t and under policy π thereafter, namely:

$$v_\pi(X_t) = E_\pi[\sum_{i=t+1}^{\infty} \gamma^{i-t-1} r_i | X_t], \ \forall X_t.$$

Algorithm 3 Global stabilization of BCN under B state-flipped control using QL method

Intput: $X_d, N, T, \tau, \epsilon$-greedy$, \omega$
Output: $\varLambda^*(X_t, X_d)$, if BCN under B state-flipped control is globally X_d stabilizable

Initialization: $Q_0(X_t, \mathcal{J}_t, X_d) \leftarrow 0, \forall X_t, \forall \mathcal{J}_t$, $E_\tau(X_d) \leftarrow \varnothing$
For $\rho = 0, 1, \cdots, N-1$ **do**
  $X_\rho \leftarrow \mathrm{rand}(\mathcal{D}^n)$
  $\alpha_\rho \leftarrow 1/(\rho+1)^\omega, t \leftarrow 0, X_t \leftarrow X_\rho$
  While $(t < T) \wedge (X_t \neq X_d)$ **do**
    Choose $\mathcal{J}_t$ using ϵ-greedy
    $\mathrm{apply}(\mathcal{J}_t), \mathrm{read}(X_{t+1}), \mathrm{read}(r_{t+1})$
    $Q_{t+1}(X_t, \mathcal{J}_t, X_d) \leftarrow Q_t(X_t, \mathcal{J}_t, X_d) + \alpha_\rho[r_{t+1} +$
    $\gamma \max\limits_{\mathcal{J}} Q_t(X_{t+1}, \mathcal{J}, X_d) - Q_t(X_t, \mathcal{J}_t, X_d)]$
    If $(X_{t+1} == X_d) \wedge (t < \tau)$ **then**
      $E_\tau(X_d) \leftarrow E_\tau(X_d) \cup X_\rho$
    end if
    $t \leftarrow t+1$
  end While
  $Q_0(X_t, \mathcal{J}_t, X_d) \leftarrow Q_t(X_t, \mathcal{J}_t, X_d), \forall X_t, \forall \mathcal{J}_t$
end for
If $E_\tau(X_d) == \mathcal{D}^n$ **then**
  $Q^*(X_t, \mathcal{J}_t, X_d) \leftarrow Q_t(X_t, \mathcal{J}_t, X_d), \forall X_t, \forall \mathcal{J}_t$
  $\varLambda^*(X_t, X_d) \leftarrow \arg\max\limits_{\mathcal{J}} Q^*(X_t, \mathcal{J}, X_d), \forall X_t$
else BCN under B state-flipped control is not globally stabilizable to X_d and discard $Q_t(X_t, \mathcal{J}_t, X_d)$
end if

The optimal policy π^* is the policy maximizing the value function at any initial state, i.e., $\pi^*(X_t, \mathcal{J}_t) = \arg\max\limits_{\pi \in \varPi} v_\pi(X_t)$, where $\varPi$ the set of all policies. Under the optimal policy π^*, the value function is denoted as $v^*(X_t) = v_\pi(X_t)$. In [25], $v_\pi(\cdot)$ satisfies the Bellman optimality equation as follows:

$$v^*(X_t) = \max_{\mathcal{J}} \sum_X \mathrm{P}\{X|X_t, \mathcal{J}\}[E[r_{t+1}|X_t, \mathcal{J}] + \gamma v^*(X)],$$

where $\mathrm{P}\{X|X_t, \mathcal{J}\}$ is the conditional probability for X_t to X by taking the joint control pair $\mathcal{J}$. Similarly, we set the action-value function $q_\pi(X_t, \mathcal{J}_t)$ to be the expected return from X_t, under $\mathcal{J}_t$ based on policy

π, i.e., $q_\pi(X_t,\mathcal{J}_t)=E_\pi[r_{t+1}+\gamma v_\pi(X_{t+1})]$. Accordingly, the optimal action-value function is defined as $q^*(X_t,\mathcal{J}_t):=q_{\pi^*}(X_t,\mathcal{J}_t),\forall X_t,\mathcal{J}_t$. Since $v^*(X_t)=\max\limits_{\mathcal{J}} q^*(X_t,\mathcal{J})$, according to the Bellman optimality equation of $v^*(X_t)$, we can obtain that

$$q^*(X_t,\mathcal{J}_t)=\sum_X \mathrm{P}\{X|X_t,\mathcal{J}\}[E[r_{t+1}|X_t,\mathcal{J}_t]+ \gamma\max_{\mathcal{J}} q^*(X,\mathcal{J})]. \tag{12}$$

π is called deterministic policy if it allows only one action for each state, i.e., with the form $\lambda(X_t)$, that maps states X_t into actions $\mathcal{J}_t=\lambda(X_t),\forall X_t$. Further, for any MDP, there exists an optimal policy which is not worse than any other policy [25], and under the optimal policy, the actions $\lambda^*(X_t)$ can be derived as

$$\lambda^*(X_t)=\arg\max_{\mathcal{J}} q^*(X_t,\mathcal{J}),\ \forall X_t.$$

In [19], temporal difference (TD) learning is introduced to tackle (12), and Q factor is introduced to be an estimation of $q_\pi(X_t,\mathcal{J}_t)$. Its iterative equation is $Q_\pi(X_t,\mathcal{J}_t)=r_{t+1}+\gamma Q_\pi(X_{t+1},\lambda(X_{t+1}))$. The learned action-value function Q can be used to estimate the optimal action-value function q^* directly. It dramatically simplifies the analysis of the algorithm and obtains the proofs of convergence of the algorithm. Then, we define TD error as $TD_{t+1}=r_{t+1}+\gamma Q_\pi(X_{t+1},\lambda(X_{t+1}))-Q_\pi(X_t,\mathcal{J}_t)$. Next, Q can update with the rule in the following:

$$Q_{t+1}(X_t,\mathcal{J}_t,X_d)=Q_t(X_t,\mathcal{J}_t,X_d)+\alpha_t[TD_{t+1}],$$
$$TD_{t+1}=r_{t+1}+\gamma\max_{\mathcal{J}} Q_t(X_{t+1},\mathcal{J},X_d)-Q_t(X_t,\mathcal{J}_t,X_d),$$

where Q table is in $\mathbb{R}_{2^n\times 2^n}$. Recalling conditions i) and ii), Q table is convergent and converges to Q^* table. Thus, for any initial state X_0, we can use Q^* table to estimate q^*, and hence we can find the optimal state-flipped transitions to X_d. Finally, we can obtain the optimal joint control pair sequence $\Lambda^*(X_0,X_d)$.

After introducing the update rule of Q factor, we continue to introduce other necessary notations in Algorithm 3: Each episode $\rho\in[0,N-1]$ is a complete training process from any initial state X_0 to the target state X_d, where N is the maximal number of episodes we consider. ϵ-greedy strategy is a common algorithmic idea, which refers to choosing the action $\mathcal{J}_t$ with the largest Q_t in the current view by probability $1-\epsilon$, i.e. $\mathcal{J}_t=\arg\max Q_t$. With probability ϵ, the choice of the action is random. In each episode ρ, we denote T as the maximum of actions taken by the agent. We set $\tau=2^n-1$ and $T\gg\tau$. In addition, we denote $E_\tau(X_d)$ as the set of states which arrive to X_d after (within) τ state-flipped transition steps.

4 Simulations

In this section, a simple BCN is used to demonstrate the obtained theoretical results.

Example 3 Reconsider BCN (1), the state transition matrix of BCN (9) is given by

$$G=\delta_8[2\ 1\ 1\ 5\ 5\ 2\ 1\ 7\ 1\ 2\ 1\ 5\ 5\ 1\ 1\ 7].$$

Then, we can obtain that

$$M=\sum_{q=1}^{2}G_q=[\delta_8^{1,2}\ \delta_8^{1,2}\ 2\delta_8^1\ 2\delta_8^5\ 2\delta_8^5\ \delta_8^{1,2}\ 2\delta_8^1\ 2\delta_8^7].$$

There are 3 fixed points $\delta_8^1,\delta_8^2,\delta_8^5$ in traditional BCN (9) without state-flipped control. However, by calculating M^8, we can obtain that $M^8=[128\delta_8^{1,2}\ \ 128\delta_8^{1,2}\ \ 128\delta_8^{1,2}\ \ 256\delta_8^5\ \ 256\delta_8^5\ \ 128\delta_8^{1,2}\ \ 128\delta_8^{1,2}\ \ 128\delta_8^{1,2}]$, which means that BCN (9) cannot achieve global stabilization only by free control sequences. Fig. 1 depicts the state transition graph of BCN (9). Now, we consider whether some state-flipped controls can be added for stabilization. Let the target state be $x_d=\delta_8^7$. Regardless of the reality constraint, give the initial flip set $B=\{1,2,3\}$. Using Algorithm 1, we can obtain that $(M\mathcal{C}_{\{2,3\}})_{7,7}=2>0$ and $\mathrm{Row}_7(M\mathcal{C}_{\{2,3\}})^2>0$. Based on Theorem 3, BCN (9) under $\{2,3\}$ state-flipped control is globally stabilizable to δ_8^7. It also implies that the stabilizing kernel is $B_{\gamma_i}=B_{2_3}=\{2,3\}$, and the corresponding stabilizing step is 2. Based on the Definition 9 of k step reachable set, using $\widetilde{G}=M\mathcal{C}_{\{2,3\}}$, we can obtain that $E_1(\delta_8^7)=\{\delta_8^5,\ \delta_8^6,\ \delta_8^7,\ \delta_8^8\}$, $E_1(\delta_8^7)=\{\delta_8^1,\ \delta_8^2,\ \delta_8^3,\ \delta_8^4\}$. There exists $N=2$, $\bigcup\limits_{k=1}^{2}E_k(x_d)=\Delta_8$. Hence, Theorem 4 can be also used to check stabilization.

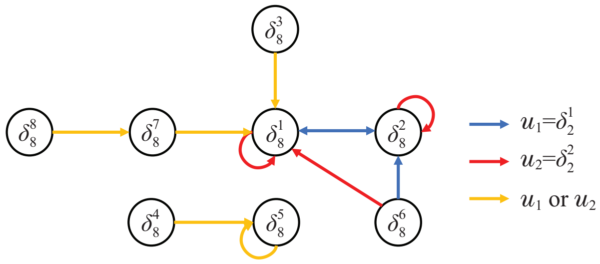

Fig. 1 The state transition graph of BCN (9)

Next, we adopt the QL Algorithm 3 to find the joint control pair sequences to steer the BCN under $\{2,3\}$ state-flipped transition to achieve global stabilization to $x_d=\delta_8^7$. Set the reward by (11), and let $N=500000$, $\omega=0.51$, $\epsilon=0.3$. The converged Q^* table can be obtained:

$$Q^*=\begin{bmatrix}64&64&64&64&64&64&80&80\\64&64&80&80&64&64&64&64\\64&64&64&64&80&80&64&64\\80&80&64&64&64&64&64&64\\80&80&64&64&64&64&100&100\\64&64&100&100&80&80&64&64\\0&0&0&0&0&0&0&0\\100&100&64&64&64&64&80&80\end{bmatrix}, \tag{13}$$

where each column in the Q table represents the joint control pair, which is $(\eta_{\emptyset}^{\neg},\ \delta_2^1)$, $(\eta_{\emptyset}^{\neg},\ \delta_2^2)$, $(\eta_{\{2\}}^{\neg},\ \delta_2^1)$, $(\eta_{\{2\}}^{\neg},\ \delta_2^2)$, $(\eta_{\{3\}}^{\neg},\ \delta_2^1)$, $(\eta_{\{3\}}^{\neg},\ \delta_2^2)$, $(\eta_{\{2,3\}}^{\neg},\ \delta_2^1)$, $(\eta_{\{2,3\}}^{\neg},\ \delta_2^2)$ from left to right, respectively. Then, based on (13), we have the optimal policy Λ^* (joint control pair sequence) for any initial state. To improve readability, we denote them by paths:

$$P_1=\{x_0=\delta_8^1 \xrightarrow[(\eta_{\{2,3\}}^{\neg},\delta_2^2)]{(\eta_{\{2,3\}}^{\neg},\delta_2^1)} x_1=\delta_8^5 \xrightarrow[(\eta_{\{2,3\}}^{\neg},\delta_2^2)]{(\eta_{\{2,3\}}^{\neg},\delta_2^1)} x_2=\delta_8^7\},$$

$$P_2=\{x_0=\delta_8^2 \xrightarrow[(\eta_{\{2\}}^{\neg},\delta_2^2)]{(\eta_{\{2\}}^{\neg},\delta_2^1)} x_1=\delta_8^5 \xrightarrow[(\eta_{\{2,3\}}^{\neg},\delta_2^2)]{(\eta_{\{2,3\}}^{\neg},\delta_2^1)} x_2=\delta_8^7\},$$

$$P_3=\{x_0=\delta_8^3 \xrightarrow[(\eta_{\{3\}}^{\neg},\delta_2^2)]{(\eta_{\{3\}}^{\neg},\delta_2^1)} x_1=\delta_8^5 \xrightarrow[(\eta_{\{2,3\}}^{\neg},\delta_2^2)]{(\eta_{\{2,3\}}^{\neg},\delta_2^1)} x_2=\delta_8^7\},$$

$$P_4=\{x_0=\delta_8^4 \xrightarrow[(\eta_{\emptyset}^{\neg},\delta_2^2)]{(\eta_{\emptyset}^{\neg},\delta_2^1)} x_1=\delta_8^5 \xrightarrow[(\eta_{\{2,3\}}^{\neg},\delta_2^2)]{(\eta_{\{2,3\}}^{\neg},\delta_2^1)} x_2=\delta_8^7\},$$

$$P_5=\{x_0=\delta_8^5 \xrightarrow[(\eta_{\{2,3\}}^{\neg},\delta_2^2)]{(\eta_{\{2,3\}}^{\neg},\delta_2^1)} x_1=\delta_8^7\},$$

$$P_6=\{x_0=\delta_8^6 \xrightarrow[(\eta_{\{2\}}^{\neg},\delta_2^2)]{(\eta_{\{2\}}^{\neg},\delta_2^1)} x_1=\delta_8^7\},$$

$$P_7=\{x_0=\delta_8^7 \xrightarrow[(\eta_{\{3\}}^{\neg},\delta_2^2)]{(\eta_{\{3\}}^{\neg},\delta_2^1)} x_1=\delta_8^7\},$$

$$P_8=\{x_0=\delta_8^8 \xrightarrow[(\eta_{\emptyset}^{\neg},\delta_2^2)]{(\eta_{\emptyset}^{\neg},\delta_2^1)} x_1=\delta_8^7\}.$$

The joint control pairs above and below the arrow are both allowed in the state-flipped transition between two states. Fig. 2 shows the state-flipped transitions considering all joint control pairs of BCN (9) under $\{2,3\}$ state-flipped transition. For two states, we take any feasible joint control pair composing a state-flipped transition graph of BCN (9) under $\{2,3\}$ state-flipped control, which is shown in Fig. 3. Comparing Fig. 1 and Fig. 3, note that although BCN (9) is not globally stabilizable by free control sequences, BCN (9) under $\{2,3\}$ state-flipped control is globally stabilizable to the target state δ_8^7 after adding state-flipped control.

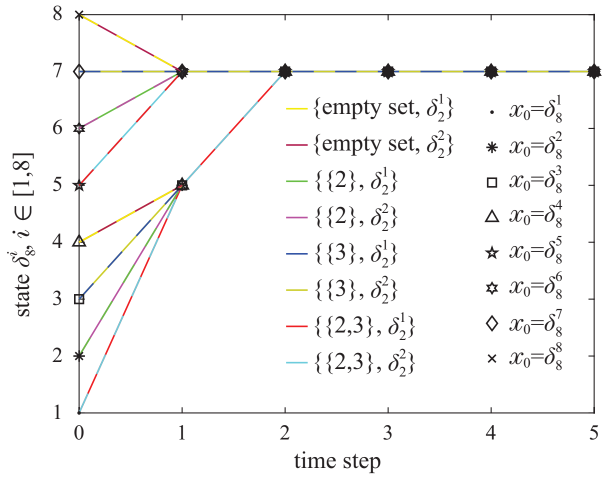

Fig. 2 All paths about state-flipped transitions of BCN (9) under $\{2,3\}$ state-flipped control

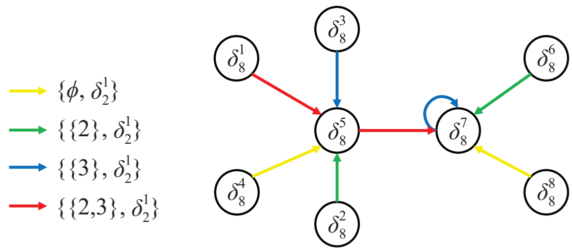

Fig. 3 One of the state-flipped transition graphs of BCN (9) under $\{2,3\}$ state-flipped control

5 Conclusion

This paper addresses the global stabilization of BCNs under state-flipped control. We propose a BCN added with state-flipped control, called BCN under state-flipped control. The state-flipped-transition matrix is given to judge the reachability of states. Based on the state-flipped-transition matrix, several criteria are proposed for the global stabilization. We design an algorithm for finding a stabilizing kernel and the corresponding stabilizing step. Moreover, a QL algorithm is given for finding the joint control pair sequences to achieve global stabilization. Finally, an example is provided to illustrate the main results.

References:

[1] KAUFFMAN S. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 1969, 22(3): 437 – 467.

[2] ZHANG Y, LIU Y, YANG X, et al. A reset algorithm solving coordination with antagonistic reciprocity. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2021, DOI: 10.1109/TSMC.2021.3049149.

[3] CHENG D, QI H, LI Z. *Analysis and Control of Boolean Networks: A Semi-tensor Product Approach*. London: Springer, 2011.

[4] CHENG D, QI H. Controllability and observability of Boolean control networks. *Automatica*, 2009, 45(7): 1659 – 1667.

[5] ZHAO Y, QI H, CHENG D. Input-state incidence matrix of Boolean control networks and its applications. *Systems and Control Letters*, 2010, 59(12): 767 – 774.

[6] LI R, YANG M, CHU T. State feedback stabilization for Boolean control networks. *IEEE Transactions on Automatic Control*, 2013, 58(7): 1853 – 1857.

[7] YU Y, WANG B, FENG J. Input observability of Boolean control networks. *Neurocomputing*, 2019, 333(14): 22 – 28.

[8] CHENG D, LI Z, QI H. Canalizing Boolean mapping and its application to disturbance decoupling of Boolean control networks. *Proceedings of 2009 IEEE International Conference on Control and Automation*. Piscataway: IEEE, 2009: 7 – 12.

[9] FORNASINI E, VALCHER M E. Fault detection of Boolean control networks. *Proceedings of the 53rd IEEE Conference on Decision and Control (CDC2014)*. Piscataway: IEEE, 2014: 6542 – 6547.

[10] LI X, LI H, ZHAO G. Function perturbation impact on feedback stabilization of Boolean control networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2019, 30(8): 2548 – 2554.

[11] ZHONG J, LU J, LIU Y, et al. Synchronization in an array of output-coupled Boolean networks with time delay. *IEEE Transactions on Neural Networks and Learning Systems*, 2014, 25(12): 2288 – 2294.

[12] ZHONG J, HO D, LU J. A new approach to pinning control of Boolean networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2021, DOI: 10.1109/TCNS.2021.3106453.

[13] GUO Y, WANG P, GUI W, et al. Set stability and set stabilization of Boolean control networks based on invariant subsets. *Automatica*, 2015, 61: 106 – 112.

[14] LU J, LIU R, LOU J, et al. Pinning stabilization of Boolean control networks via a minimum number of controllers. *IEEE Transactions on Cybernetics*, 2021 51(1): 373 – 381.

[15] RAFIMANZELAT M, BAHRAMI F. Attractor controllability of Boolean networks by flipping a subset of their nodes. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 2018, 28(4): 43120.

[16] RAFIMANZELAT M, BAHRAMI F. Attractor stabilizability of Boolean networks with application to biomolecular regulatory networks. *IEEE Transactions on Control of Network Systems*, 2019, 6(1): 72 – 81.

[17] ZANG Q, FENG J, ZHAO Y, et al. Stabilization and set stabilization of switched Boolean control networks via flipping mechanism. *Nonlinear Analysis: Hybrid Systems*, 2021, DOI: 10.1016/j.nahs.2021.101055.

[18] LIU Z, ZHONG J, LIU Y, et al. Weak stabilization of Boolean networks under state-flipped control. *IEEE Transactions on Neural Networks & Learning Systems*, 2021, DOI: 10.1109/TNNLS.2021.3106918.

[19] WATKINS C J, DAYAN P. Q-learning. *Machine Learning*, 1992, 8(3/4): 279 – 292.

[20] ZHANG Q, PAN W, REPPA V. Model-reference reinforcement learning for collision-free tracking control of autonomous surface vehicles. *Proceedings of the 59th IEEE Conference on Decision and Control (CDC)*. Piscataway: IEEE, 2020, DOI: 10.1109/CDC42340.2020.9304347.

[21] TORRES P J R, GARCIA C G, IZQUIERDO S K. *Reinforcement Learning with Probabilistic Boolean Network Models of Smart Grid Devices*, 2021, arxiv: 2102.01297.

[22] ZHOU C, LIU S, LI X, et al. An intelligent traffic signal control system based on deep reinforcement learning. *Proceedings of 2020 Information Communication Technologies Conference*. Piscataway: IEEE, 2020, DOI: 10.1109/ICTC49638.2020.9123260.

[23] ACERNESE A, YERUDKAR A, GLIELMO L, et al. Reinforcement learning approach to feedback stabilization problem of probabilistic Boolean control networks. *IEEE Control Systems Letters*, 2021, 5(1): 337 – 342.

[24] ACERNESE A, YERUDKAR A, GLIELMO L, et al. Model-free self-triggered control co-design for probabilistic Boolean control networks. *IEEE Control Systems Letters*, 2021, 5(5): 1639 – 1644.

[25] SUTTON R, BARTOA. *Reinforcement Learning: An Introduction*. Cambridge: MIT press, 2018.

作者简介:

刘 洋 教授, 博士生导师, 目前研究方向为逻辑系统、混杂系统与分布式优化, E-mail: liuyang@zjnu.edu.cn;

刘泽娇 硕士研究生, 目前研究方向为逻辑系统控制与强化学习, E-mail: liuzejiao@zjnu.edu.cn;

卢剑权 教授, 博士生导师, 目前研究方向为多智能体系统控制与复杂网络, E-mail: jqluma@seu.edu.cn.