Online Observability of Boolean Control Networks*

Guisen Wu Liyun Dai* Zhiming Liu
RISE & School of Computer and Information Science, Southwest University Chongqing, China
{wgs233,dailiyun,zhimingliu88}@swu.edu.cn

Taolue Chen
Department of Computer Science and Information Systems Birkbeck, University of London
taolue@dcs.bbk.ac.uk

Jun Pang
Faculty of Science, Technology and Communication University of Luxembourg
jun.pang@uni.lu

Hongyang Qu
Department of Automatic Control and Systems Engineering University of Sheffield
h.qu@sheffield.ac.uk

***Abstract*—Four types of observability of Boolean control networks (*BCNs*) have been proposed to solve different problems. However, all of them are offline observabilities that we can not adjust the input sequence by observing the output sequence in the process of determining the initial state of *BCNs*. In this paper, we firstly propose the online observability that it can determine the initial state of *BCNs* dynamically in real time. The online observability accomplishes this task by deciding input sequence and observing out sequence in every time step. As the node values of *BCNs* update at discrete time, we use the time step to represent the discrete time. Compared with offline observabilities, only the online obsevability can help us to determine the initial state of some biological systems which can be checked at most once.**

***Index Terms*—Boolean control networks, online observability, directed graph, supertree**

I. INTROCTION

In 1960s, Nobel Prize winners Jacob and Monod found that “Any cell contains a number of ‘regulatory’ genes that act as switches and can turn one another on and off. If genes can turn one another on and off, then you can have genetic circuits.” [1], [2]. Inspired by these Boolean-type actions in genetic circuits, the Boolean networks (*BNs*) is firstly proposed by Kauffman [3] for modeling nonlinear and complex biological systems. Some general descriptions of the *BNs* and its applications to biological systems can be found in Kauffman. Since then research interests in *BNs* have been motivated by the large number of natural and artificial systems. These natural and artificial systems describing variables display only two distinct configurations, and hence these describing variables take only two values, i.e., $\{0, 1\}$ [4], [5], [6], [7], [8], [9].

When extenal regulation or perturbation is considered, *BNs* are naturally extended to Boolean control networks (*BCNs*) [10]. The *BCNs* can be used to solve various important realistic problems. For instance, first *BCNs* can be used to do structural and functional analysis of signaling and regulatory networks [11], [12]. Second *BCNs* can be used for abduction based drug target discovery [13]. Furthermore, *BCNs* also can be used for pursuit evasion problems in polygonal environments [14]. For a better understanding, we make a brief introduction about how *BCNs* can be used to do structural and functional analysis of signaling and regulatory networks. Evolution has equipped cells with exquisite signaling systems which allow them to sense their environment. The immune system is a very important part of the signaling system, it can identify and eliminate foreign invading antigens. T-cells known as lymphocytes are a type of white blood cells, they play a central role in the immune system. T-cells can recognize protentially dangerous agents for cells and initiate an reaction against these agents. T-cells do so by T-cell receptors to detect foreign antigens bound to major histocompatiblity complex molecules, and then activate, through a signaling cascade, several transcription factors. As the interaction of the antigen-specific receptor of T-cells with its antigenic ligand can lead either to cell activation (1) or to a state of profound unresponsiveness (0). We would like to apply the *BCNs* to study the T-cell receptor kinetics model better [11], [12]. The network graph of the *BCN* T-cell receptor kinetics model given in [12] is shown in Fig.1. For more details, we refer the reader to read [12]. Furthermore, there is an approach to study large-scale *BCNs* via network aggregations [20]. However, in order to further improve the performance of the *BCN* model, we make some optimizations about the definition of observability of *BCNs*.

As the wide application of *BCNs*, there are much research work about the control-theoretic problems of *BCNs*. The study on control-theoretic problems of *BCNs* can date back to 2007 [15]. The work above also proves that the problem of determining the controllability of *BCNs* is *NP*-hard in the number of nodes. Furthermore, it points out that “One of the major goals of systems biology is to develop a control theory for complex biological systems.” Since then, the study on control-theoretic problems in the areas of *BNs* and *BCNs* has drawn great attention [2], [16], [17], [18], [9]. Besides controllability, observability is an another basic control-theoretic problems and it also attract many attentions. Among these studies, *semi-tensor product* (STP) of matrices is one of useful tool to deal with both *BNs* and *BCNs* related problems [2]. Moreover, [2] gives equivalent conditions for controllability of *BCNs* and observability of controllable *BCNs*. To date, there are four types of observability have been proposed.

*Corresponding author

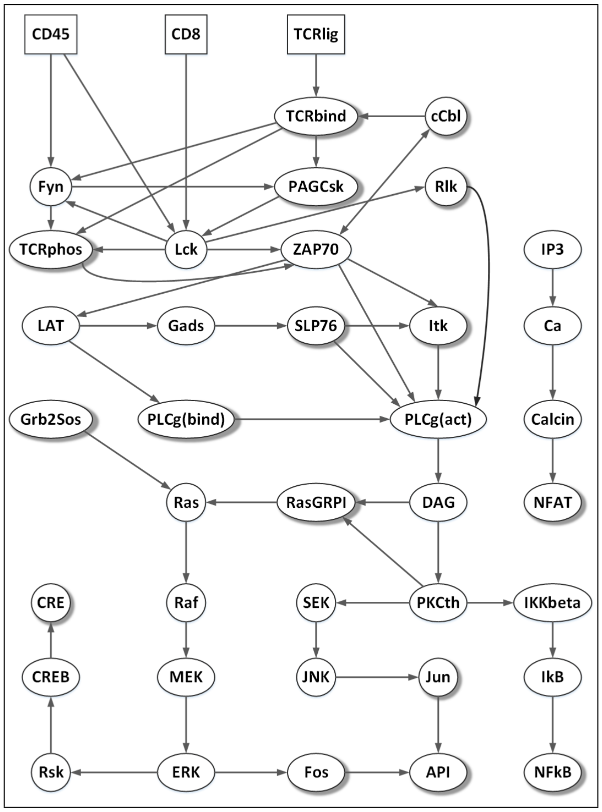

Fig. 1. Network graph of the T-cell receptor kinetics model, where rectangles denote input nodes, the other nodes denote state nodes, particularly the nodes with shadows are chosen to be observed.

1) The first type of observability proposed in 2009 [2] means that every initial state can be determined by an input sequence.
2) The second observability proposed in 2010 [16] stands that for every two distinct initial states, there exists an input sequence which can distinguish them, and this observability is determined in [19].
3) The third observability proposed in 2011 [17] states that there is an input sequence that determines the initial state.
4) The fourth observability proposed in 2013 [9] is essentially the observability of linear control systems, i.e., every sufficient long input sequence can determine the initial state.

In above mentioned definitions an input is not the value of an input-node of the *BCN*, but it represents the values of all input-nodes of the *BCN* on a time step. Therefore, an input can be seen as a vector of the values of all input-nodes of the *BCN* on a time step. An input sequence consists of several inputs in sequential time steps. A output can also be seen as a vector of the values of all output-nodes of the *BCN* in a time step. In every time step, there is a pair of input and output of *BCN*. Therefore a output sequence also consists of several outputs in sequential time steps. In *Section II*, we will list the informal definition of four offline observabilities as well as the formal definition of four observabilities.

The four types of observability have many nice properties that they can be used in some useful applications. However, all of four types of observability of *BCNs* are offline observabilities which means that they can not adjust the input sequence by observing the output sequence in the process of determining the initial state of *BCNs*. Therefore, we propose the online observability that we can determine the initial state of *BCNs* dynamically. In other words, the online observability decides the input sequence in each time step by observing the out sequence. In the online observability, we infer the possible initial states set by observe outputs of *BCN* in the first k time steps. Through the possible initial states, we can choose one input to refine the possible initial states set in the time step $k+1$. Repeat above procedure until the cardinality of initial states set turns into be one then we can determine the initial state of *BCNs*. That is why we call this process a dynamic process.

Contribution: Firstly, we propose the concept of the online observability of *BCNs*. Compared with four existing observabilities, the online observability can help us to determine the initial state of some biological systems which can be checked at most once. Secondly, in addition to theoretical research, we also provide two algorithms to determine the online observability for *BCNs*. Finally, we introduce some applications of the online observability of *BCNs*. Including takes less observation costs, methods to find shortest path and approaches to avoid entering critical states when we use it to determine the initial state of *BCNs*. These applications will explain the advantages of online observability of *BCNs* comparing with offline observabilities.

Structure: The remainder of this paper is organized as follows. *Section II* introduces necessary preliminaries about *BCNs*, algebraic forms of *BCNs* and the four existing kinds of observability of *BCNs*. *Section III* presents the definition of deduction function, k steps determinability and online observability of *BCNs*. *Section IV* presents how to determine the online observability of *BCNs* by super tree and directed graph. *Section V* talks about some applications of the online observability of *BCNs*. We also compare the online observability with offline observabilities in this section. *Section VI* ends up with the introduction of our future work.

II. Preliminaries

In this section we introduce the definition of *BCNs* and their algebraic form as well as the four existing kinds of observability of *BCNs*.

A. Boolean Control Networks

A Boolean control network can be described as a directed graph together with logical equations to describe the updating rules of the nodes of this directed graph, the definition of *BCN* is as follows.

Definition 1: ([10]) A *BCN* consists of input-nodes, state-nodes, output-nodes, and directed edges which connect nodes. A node in *BCN* can take a logic value from $\{0, 1\}$ at a discrete time $0, 1, 2, \ldots$ For one directed edge from a state-node s_1 (or an input-node i_1) to a state-node s_2 means that the logic value of s_2 at time step $t+1$ is affected by the value of s_1 (or i_1) at time step t. For one directed edge from a state-node s_1 to a output-node o_1 means that the logic value of o_1 at time step t is affected by the value of s_1 at time step t.

Note that one can only know that whether a node is affected by another node from the network graph. Different *BCNs* may have the same structure, in order to determine a *BCN* uniquely, logical equations are also needed to describe the specific updating rules of *BCNs*.

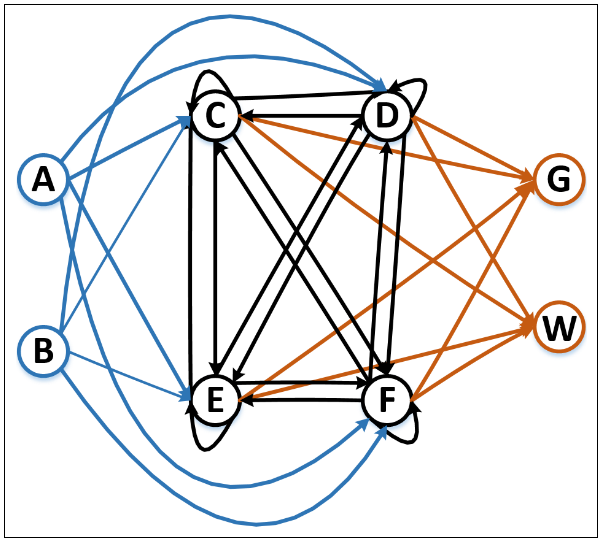

Fig. 2. A Boolean control network with two input-nodes A and B, four state-nodes C, D, E and F, and two output-nodes G, W. We use blue, black and orange, to distinguish three kinds of nodes and three kinds of edges.

To better illustrate the concept of *BCNs*, we give a simple example to describe it.

Example 1: In Fig.2 we have a *BCN* with two input-nodes A and B, four state-nodes C, D, E and F, and two output-nodes G, W. The *BCN* is shown in Fig.2, and the updating rules of this *BCN* are described as truth table Fig.3. The reason why we use truth table to describe the updating rules of the *BCN* is that this form of updating rules will be more convenient for *BCN* to be converted into its aglebraic form. What's more, for instance, the updating rule of state-node G is that,

$$G(t) = C(t) \wedge \neg(\neg D(t) \wedge \neg E(t) \wedge \neg F(t))$$

For convenience, we will use this example in the whole paper to explain various concepts we introduce.

B. The algebraic forms of BCNs

To better illustrate the concept of algebraic forms, in this paper, we investigate the *BCN* in the following. And we suppose that this *BCN* has n state-nodes, m input-nodes and q output-nodes. Then the updating rules of the *BCN* can be described as following formulas:

$$\begin{aligned} s(t+1) =& f(i(t), s(t)) \\ o(t) =& h(s(t)) \end{aligned} \tag{1}$$

Truth Table																	A(t)	B(t)
C(t)	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0		
D(t)	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0		
E(t)	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0		
F(t)	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0		
C(t+1):	0	1	0	0	0	1	1	1	0	1	1	0	1	1	1	0	1	1
	1	0	0	0	0	1	1	0	1	0	0	0	1	1	1	1	1	0
	1	1	1	0	0	0	1	1	0	0	1	0	0	0	1	0	0	1
	1	0	1	1	0	1	0	0	1	1	0	1	0	0	0	0	0	0
D(t+1):	1	1	1	0	1	0	1	0	0	1	1	1	0	0	0	0	1	1
	0	1	0	1	0	1	1	0	1	0	1	0	0	0	1	0	1	0
	0	1	1	0	0	1	0	1	0	0	0	0	1	1	1	0	0	1
	1	0	0	0	1	0	1	1	0	0	0	1	1	1	0	0	0	0
E(t+1):	1	0	0	0	1	1	1	0	0	1	0	0	0	1	0	1	1	1
	0	1	0	1	1	0	0	0	1	1	0	1	1	0	1	1	1	0
	0	1	0	1	1	1	1	1	0	1	1	1	0	1	0	0	0	1
	1	1	0	1	1	0	0	0	1	1	1	0	1	0	0	0	0	0
F(t+1):	0	0	1	0	1	1	1	1	1	0	1	0	1	0	0	1	1	1
	0	1	1	0	0	0	1	0	1	0	0	1	1	1	0	0	1	0
	1	0	1	1	1	1	1	1	0	1	0	0	1	0	0	1	0	1
	1	0	1	0	1	0	1	0	1	1	1	1	0	0	0	0	0	0
G(t):	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0		
W(t):	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	0		

Fig. 3. The truth table which describe the updating rules of the *BCN* shown in *Fig.2*.

$s(t) \in \mathbb{B}^n$ are state-nodes; $i(t) \in \mathbb{B}^m$ are input-nodes; $o(t) \in \mathbb{B}^q$ are output-nodes; $f : \mathbb{B}^{n+m} \mapsto \mathbb{B}^n$ and $h : \mathbb{B}^n \mapsto \mathbb{B}^q$ are logical functions that represent the updating rules of *BCNs*. Where $\mathbb{B}$: the set $\{0, 1\}$; $t = 0, 1, \ldots$ represents the discrete time.

Therefore in the previously mentioned example, we have that $C(t), D(t), E(t), F(t) \in s(t)$; $A(t), B(t) \in i(t)$ and $G(t), W(t) \in o(t)$; $n = 4$, $m = 2$ and $q = 2$; f and h are described in the truth table (*Fig.3*).

Furthermore, the *STP* of matrices can be used to represent the algebraic forms of *BCNs* [2], the definition of *STP* is as follows.

Definition 2 (STP): [18] Let $X \in \mathbb{R}_{m\times n}$, $Y \in \mathbb{R}_{p\times q}$ and $\alpha = lcm(n, p)$ be the least common multiple of n and p. The STP of X and Y is defined as

$$X \ltimes Y = (X \otimes I_{\alpha/n})(Y \otimes I_{\alpha/p}),$$

where $\otimes$ denotes the Kronecker product.

After introducing the definition of *STP* of matrices, we introduce some related notations at first [21]:

- δ_n^i: the i-th column of the identity matrix I_n;
- Δ_n: the set $\{\delta_n^1, \ldots, \delta_n^n\}$;
- $\delta_n\,[i_1, \ldots, i_s]$: $\left[\delta_n^{i_1}, \ldots, \delta_n^{i_s}\right]$ $(i_1, \ldots, i_s \in \{1, 2, \ldots, n\})$ the logical matrix;
- $L_{n\times s}$: the set of $n \times s$ logical matrices.

Using *STP* of matrices, the formula (1) can be quivalently represented in the following algebraic form:

$$\begin{aligned} s(t+1) =& L \ltimes i(t) \ltimes s(t) \\ o(t) =& H \ltimes s(t) \end{aligned} \tag{2}$$

where $s(t) \in \Delta_N$, $i(t) \in \Delta_M$, and $o(t) \in \Delta_Q$ denote the states, inputs and outputs respectively the same as in formula *(1)*, but $s(t)$, $i(t)$ and $o(t)$ in formula *(2)* would be written in special vector forms; $L \in L_{N\times(NM)}$ and $H \in L_{Q\times N}$ denote the relation matrices; that $N = 2^n$, $M = 2^m$, and $Q = 2^q$. Since *STP* keeps most properties of the conventional product [18], the associative law, the distributive law, etc., we usually omit the symbol "$\ltimes$" hereinafter. For instance, the formula "$s(t+1) = L \ltimes i(t) \ltimes s(t)$" will be written as "$s(t+1) = Li(t)s(t)$" in the following pages.

To construct algebraic form (2) we give a mapping $\tau : \{0,1\} \mapsto \{\delta_2^1, \delta_2^2\}$ where $\tau(0) = \delta_2^2$, $\tau(1) = \delta_2^1$. Therefore, the logical variable $A(t)$ takes value from these two vectors, i.e., $A(t) \in \{\delta_2^1, \delta_2^2\}$. Using the *STP* of matrices, we have

$$i(t) = i_1(t)\ldots i_m(t);$$

$$s(t) = s_1(t)\ldots s_n(t);$$

$$o(t) = o_1(t)\ldots o_q(t).$$

And according to [22], each logical function f_p of state-nodes can be found in the updating rules (1). The form of f_p as:

$$f_p(i_1(t), \ldots, i_m(t), s_1(t), \ldots, s_n(t))$$

and there exists a structure matrix $L_p \in L_{2\times NM}$ such that

$$\tau(f_p(i_1(t), \ldots, i_m(t), s_1(t), \ldots, s_n(t))) = L_p i(t)s(t) \quad (3)$$

For state-nodes $s_1, \ldots, s_n$, we have n logical matrices $L_1, \ldots, L_n$ for them, respectively. If for each state-node s_p the logical matrix has its form

$$L_p = [\delta_2^{p_1}, \ldots, \delta_2^{p_{NM}}],$$

then we have that

$$L = [\delta_N^{R_1}, \ldots, \delta_N^{R_{NM}}]$$

where

$$\delta_N^{R_1} = \delta_2^{1_1} \ldots \delta_2^{n_1}; \ldots; \delta_N^{R_{NM}} = \delta_2^{1_{NM}} \ldots \delta_2^{n_{NM}}.$$

By this relationship we can construct the L for the algebraic forms of *BCNs*. What's more we can also construct the logical matrix H in the similar way. To better illustrate the concept of algebraic forms, we give a simple example to describe it.

Example 2: For instance, the *BCN* whose structure is depicted in Fig.2, and the updating rules of this *BCN* is described as truth table in Fig.3. We have that the updating rules of this *BCN* can be represented with the algebraic form:

$$\begin{aligned} s(t+1) =& \delta_{16}[\alpha]i(t)s(t) \\ o(t) =& \delta_4[\beta]s(t) \end{aligned} \quad (4)$$

where $\alpha = \{10, 4, 11, 16, 9, 5, 1, 7, 15, 2, 3, 12, 7, 6, 8, 13, 8, 9, 15, 10, 14, 4, 3, 16, 1, 14, 12, 13, 5, 7, 2, 6, 7, 2, 3, 13, 13, 9, 5, 1, 16, 13, 6, 14, 11, 10, 4, 15, 1, 14, 7, 6, 9, 8, 11, 12, 5, 5, 13, 3, 10, 12, 16, 16\}$, $\beta = \{1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4\}$, $t \in \mathbb{N}$, $s \in \Delta_{16}$, $i \in \Delta_4$ and $o \in \Delta_4$.

C. *Four existing observability of* BCNs

In this subsection we introduce four existing kinds of observability of *BCNs*. Let Δ_N, Δ_M, Δ_Q be three alphabets, for all $s_0 \in \Delta_N$ and all $p \in \mathbb{Z}_+$; ∞ is the infinite natural numbers. In order to introduce four existing kinds of observability of *BCNs*, we define the mappings [21]:

$$\begin{aligned} L_{s_0}^p &: (\Delta_M)^p \mapsto (\Delta_N)^p, i_0 \ldots i_{p-1} \mapsto s_1 \ldots s_p \\ L_{s_0}^\infty &: (\Delta_M)^\infty \mapsto (\Delta_N)^\infty, i_0 i_1 \ldots \mapsto s_1 s_2 \ldots \end{aligned} \quad (5)$$

$$\begin{aligned} (HL)_{s_0}^p &: (\Delta_M)^p \mapsto (\Delta_Q)^p, i_0 \ldots i_{p-1} \mapsto o_1 \ldots o_p \\ (HL)_{s_0}^\infty &: (\Delta_M)^\infty \mapsto (\Delta_Q)^\infty, i_0 i_1 \ldots \mapsto o_1 o_2 \ldots \end{aligned} \quad (6)$$

For all $p \in \mathbb{Z}_+$, all $I = i_1 \ldots i_p \in (\Delta_M)^p$, and all $1 \geq p \geq j \geq |I|$, we use I[p,j] to denote the word $i_p \ldots i_j$ as a input sequence. Then four existing kinds of observability of BCNs can be define as:

Definition 3: The first kind of observability is that, *BCN* is called observable, if for every initial state $s_0 \in \Delta_N$, there exists an input sequence $I \in (\Delta_M)^p$ for some $p \in \mathbb{Z}_+$ such that for all states $s_0 \neq s'_0 \in \Delta_N$, $Hs_0 = Hs'_0$ implies $(HL)_{s_0}^p(I) \neq (HL)_{s'_0}^p(I)$ [2].

Hence the first observability means that if a *BCN* is observable then every initial state of the *BCN* can be determined by an input sequence. But we can only use the corresponding input sequence of a state to check whether this state is the initial state of the *BCN* or not.

Definition 4: The second kind of observability is that a *BCN* is called observable if for any distinct states s_0, $s'_0 \in \Delta_N$, there exists an input sequence $I \in (\Delta_M)^p$ for some $p \in \mathbb{Z}_+$, such that $Hs_0 = Hs'_0$ implies $(HL)_{s_0}^p(I) \neq (HL)_{s'_0}^p(I)$ [16].

The second observability means that a *BCN* is called observable if for every two distinct initial states of the *BCN*, there exists an input sequence which can distinguish them.

Definition 5: The third kind of observability is that, a *BCN* is called observable, if there exists an input sequence $I \in (\Delta_M)^p$ for some $p \in \mathbb{Z}_+$, such that for any distinct states s_0, $s'_0 \in \Delta_N$, $Hs_0 = Hs'_0$ implies $(HL)_{s_0}^p(I) \neq (HL)_{s'_0}^p(I)$ [17].

The third observability means that a *BCN* is called observable if there is an input sequence that determines the initial state of the *BCN*.

Definition 6: The fourth kind of observability is that, *BCN* is called observable, if for any distinct states s_0, $s'_0 \in \Delta_N$, for any input sequence $I \in (\Delta_M)^\infty$, $Hs_0 = Hs'_0$ implies $(HL)_{s_0}^\infty(I) \neq (HL)_{s'_0}^\infty(I)$ [9].

The fourth observability means that a *BCN* is called observable if every sufficient long input sequence can determine the initial state of the *BCN*.

Then from the definitions of four existing kinds of observability, we know that [21]:

- the first one implies the second one;
- the third one implies the second one and first one;
- the fourth one implies the third one, second one and first one.

We can not use the first one and second one to determine the initial state of *BCNs* which can be checked at most once. For example, in the first kind observability we need to assume the initial state of a *BCN*, and then check it by corresponding input sequence of this state. If the state we assume is correct, then we can determine the initial state. But if the the assumption is not correct, we can not determine the initial state of the *BCN*. Therefore we need to check several test cases (with the same initial state) of this *BCN* untill we can determine the initial state of it. However we can use the third existing observability and fourth existing observability to determine the initial state of *BCNs* through one test case. And we need not to presuppose the initial state of *BCNs* when we use the third observability and fourth observability. But the requirements for *BCNs* are very harsh when we use the third observability and fourth observability.

In some biological systems, the initial states of them can be checked at most once, i.e., we have only one test case for them. Therefore, we can not use the first observability and second observability to determine the initial states of them in real time. And in some biological systems, it would takes many costs to check these biological systems. Hence we will spend a lot of overhead to determine the initial states of them by the first observability and second observability. Furthermore, we also can not use the third observability and fourth observability to determine the initial states of some biological systems when they can not satify the requirements of the third observability and fourth observability. With these disadvantages of four existing observabilities, we propose the online observability of *BCNs* to solve this problem.

Problem: Finding the necessary and sufficient condition of determine the initial state of *BCNs* in real time.

III. The online observability of *BCNs*

In this section we propose the online observability. We give a informal definition of it at first.

A *BCN* has online observability, if every initial state $s_0 \in \Delta_N$ can be determined in real time. In the online observability we determine the state of *BCN* by dynamically deciding input sequence and observing output sequence at every time step. And this process can be accomplished in finite time steps.

The reason why we called this kind of observability online observability is that:

- In this kind of observability, we determine the initial state of *BCNs* in real time. In other words, we can use one test case to determine the initial state of *BCNs*. We call this property *real-time* property.
- In online observability, we use the outputs we observe to adjust the input sequence at every time step. By this way, we can make full use of the outputs to determine the initial state of *BCNs*. We call this property *interactivity*.

We can determine the initial state of any test case of a *BCN* when we can use one test case to determine the initial state of it. We need *real-time* property to help us avoid repeating biological experiments. And the *interactivity* would help us to find the necessary and sufficient condition of determine the initial state of *BCNs* in real time. With the *real-time* and *interactivity*, we called this kind of observability online observability.

In the left of this section, firstly we present the definition of deduction function. Secondly we present the definition of K steps determinability. We take them as the preparations for defining the online observability. Finally, we give the formal definition of the online observability of *BCNs*.

A. Deduction function

The observability we propose can determine the initial state online. In the time setp k, we observe the output of *BCNs* at first. Through this we can infer the possible values of state-nodes and denote them by possible states set S_k. At the next step, we need to decide the input i_k which satisfies that s_i, s_j will not turn into the same state after being affected by the input i_k i.e., $Ls_ii_k \neq Ls_ji_k$ for any distinct $s_i, s_j \in S_k$. After deciding input, we can observe the new output, and then we can infer the new possible states set. The cardinal number of possible states set after we inputted will not lager than the cardinal number of possible states set before we input. If the cardinal number of possible states set turn into be 1 then we can determine the state and the initial state of *BCN*. And the reason why the cardinal number of the possible states set would decrease is shown in equation (10) and (11).

To better describe the deduction process of *BCN* initial state we mentioned before, we give a deduction function for it. The definition of this function is as follows.

Definition 7 (Deduction Function): The deduction function can be defined as $\mathtt{Ded}\,(S,i,o)$. Using this function we can get a states set $\mathtt{Ded}\,(S,i,o)$ for S after inputing i and observing o. Therefore, based on deduction process mentioned before, we have that there exists the corresponding $s(t) \in S$ of $s(t+1)$ such that

$$s(t+1) = L \ltimes i \ltimes s(t) \ when\ i \neq \varepsilon,$$

and

$$H \ltimes s(t+1) = o \ when\ o \neq \varepsilon,$$

for each element

$$s(t+1) \in \mathtt{Ded}\,(S,i,o)$$

where

- $S \in 2^{\Delta_N}$ is the possible states set;
- $i \in (\Delta_M \cup \varepsilon)$ represents the input;
- $o \in (\Delta_Q \cup \varepsilon)$ represents the output;
- $\mathtt{Ded}\,(S,i,o) \in 2^{\Delta_N}$ is the possible states set after deduction.

From the definition of deduction function, we have some equations for this function. By researching these equations, we can know the details of this function better.

$$\mathtt{Ded}\,(\emptyset,i,o) = \emptyset \tag{7}$$

Equation (7) represents that if the possible states set is an empty set $\emptyset$ then no matter what we input and observe, we can only deduce the possible set is $\emptyset$. It means that if we don't know anything about the state of a *BCN*, then we can not deduce anything no matter what we do.

$$\mathtt{Ded}\left(S, \varepsilon, \varepsilon\right) = S \tag{8}$$

For any possible states set S and we neither input anything and nor observe the output. In this case we can only deduce that the possible states set is S shown in equation (8). It means that before inputing and observing the output of *BCN* we can not know more information about this *BCN* than we used to knew.

$$\mathtt{Ded}\left(\Delta_N, \varepsilon, \delta_4^1\right) = \{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\} \tag{9}$$

Using the example mentioned before (4), when the possible states set $S = \Delta_N$, and we observe that the outputs of *BCN* is δ_4^1 before we decide input. In this case we can deduce that the possible states would be δ_{16}^1, δ_{16}^2 or δ_{16}^3 shown in equation (9).

$$\mathtt{Ded}\left(\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}, \delta_4^1, \varepsilon\right) = \{\delta_{16}^{10}, \delta_{16}^4, \delta_{16}^{11}\} \tag{10}$$

And then, if the possible states set $S = \{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$ we input δ_4^1. Before we observe the output of *BCN* we can only deduce the possible states would be δ_{16}^{10}, δ_{16}^4 or δ_{16}^{11} shown in equation (10). In other words, the cardinal number of the possible states set did not decrease before observing the output of this *BCN*.

$$\mathtt{Ded}\left(\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}, \delta_4^1, \delta_4^3\right) = \{\delta_{16}^{10}, \delta_{16}^{11}\} \tag{11}$$

But if we observe that the output of *BCN* is δ_4^3, then we can deduce that the possible state can be δ_{16}^{10} or δ_{16}^{11} shown in equation (11). Such that the cardinal number of the possible states set may decrease after observing the output of *BCN*.

$$\mathtt{Ded}\left(\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6\}, \delta_4^3, \varepsilon\right) = \{\delta_{16}^9, \delta_{16}^{13}\} \tag{12}$$

Finally if the set of possible states is $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6\}$ and the inputs is δ_4^3. Before we observe the output of *BCN* we can deduce that the possible state values can be δ_{16}^9 or δ_{16}^{13} shown in equation (12). Because both δ_{16}^4 and δ_{16}^5 are turn into be the same state δ_{16}^9 after affected by δ_4^3. And if the initial state of the *BCN* is one of them then we can not determine the initial state any more. If the cardinality number of the possible states set of one *BCN* decreased before observing its output the deduction process, then we can not deduce its initial state any more.

B. *k-step determinability*

After we difined the deduction function, we can present the definition of k-step determinability of the states set of *BCNs* and the range of k is the set of natural numbers. It may easier to difine online observability by programming language. But we would like to define its mathematical form for preciseness of concepts. However, before defining the online observability of *BCNs*, we need to difine the k-step determinability of the states set of *BCNs* at first.

Definition 8 (k-Step Determinability): When $k = 0$, a set of states S the S is 0-step deterministic if the cardinal number of this states set $|S| = 1$. When $k > 0$, a set of states S is k-step deterministic if the cardinal number of this states set $|S| > 1$, and for this set of states S there exists i_p in Δ_M such that

- $|\mathtt{Ded}\left(S, i_p, \varepsilon\right)| = |S|$, and
- for each o_j in Δ_Q such that $|\mathtt{Ded}\left(S, i_p, o_j\right)| \neq 0$ and $\mathtt{Ded}\left(S, i_p, o_j\right)$ is k'-step deterministic with $k' < k$.

And we default $k \geq 0$ when we talk about whether a states set of *BCNs* S is k-step deterministic or not.

From the definition of k-stepdeterminability we know that "$k = 0$" means that we can determine the state without choosing any input and observing output. Because if we know the cardinality number of possible states set is 1, then we can know the state of *BCNs*. Therefore, we can only discuss the case of $k = 0$ when $|S| = 1$. If $k > 0$, we have $|S| > 1$. Furthermore, the definition of k steps deterministic is defined recursively, and it need to use the definition of k ($k = 0$) steps.

Furthermore, if a states set S is k_1-step deterministic and $k_1 \leq k_2$, then S is k_2-step deterministic. But if the states set S is k_1-step deterministic and $k_1 \geq k_2$, we can not make sure whether S is k_2-step deterministic or not. Therefore you can consider the "S is k-step deterministic" as "We can determine the state of a *BCN* by this possible states set S in k steps. And we finish this determination process by deciding input sequence and observing out sequence at each time step". However, we say a possible states set S is k-step deterministic (k has no specific value) means that we can determine the state of a *BCN* by this possible states set S in finite steps in the following pages of this paper.

C. *Online observability*

After the previous preparation, we present the formal definition of the online observability. The formal definition of the online observability of *BCNs* is as follows.

Definition 9 (Online Observability of BCNs): A *BCN* is called online observable, if for every o_j in Δ_Q such that $|\mathtt{Ded}\left(\Delta_N, \varepsilon, o_j\right)| \neq 0$ and $\mathtt{Ded}\left(\Delta_N, \varepsilon, o_j\right)$ is k-step deterministic.

After defining online observability of *BCNs*, we discuss the comparison of online observability with the four existing observability. In the second existing kind of observability, we presuppose the initial state of a *BCN*, and then try to find the input sequence to distinguish it from other kinds of initial states. But the input sequence determined by the presupposed initial state may make some of other kinds of initial states turn into be the same state. Such that some of other kinds of initial states can not be distinguished anymore, and if the initial state of the *BCN* is one of them then the initial state can not be determined anymore. This problem has to be considered in the online obervability of *BCNs*. Hence the online observability implies the first existing kind of observability, and then the online observability implies the second existing kind of observability. In the third existing kind of observability, there has to exist an input sequence that can distinguish any distinct states. However in online observability we can use different input sequences to distinguish any distinct states in different states sets. Where these different states sets are classified by their corresponding output. Therefore, we have the third existing kind of observability implies the online observability of *BCNs*, then the fourth existing kind of observability implies the online observability.

When I learn the four existing kinds of observability of *BCNs*, I find that if we want determine the initial state of a *BCN* by first kind of observability, we need to guess the initial state of the *BCN* and then check it by its corresponding input sequence. If the initial state we guess is right then we can determine the initial state of this *BCN*. But if what we guess is incorrect, we need to guess the initial state and input its corresponding input sequence again and again untill we determine the initial state of the *BCN*. But if we can not repeat this process, we may can not determine the initial state of the *BCN*. Then I turn my gaze to the third observability, this kind of observability makes we can determine the initial state without presupposing the initial state. But I think if we can determine the possible states set of the *BCN* by observing the output at first, why we do not try to find corresponding input sequences for these possible states sets? And then my teacher and I talk about this thinkness and expand it into the original idea of the online observability of *BCNs*.

From the informal definition and formal definition of online observability, we can know that the necessary and sufficient condition of determine the initial state of *BCNs* in real time is the online observability of *BCNs*. With the formal definition we can find the best we like to determine the initial state of some biological systems which can be represented by *BCNs*.

IV. Determining the online observability of *BCNs*

In this paper, we propose two approaches to determine the online observability of *BCNs*. The first way is by using supertree and the second way is by using directed graph. The construction process of supertree and directed graph simulate deduction process mentioned before. We check the super tree based on the definition of online observability of *BCNs* depth first or breadth first. When we find enough leaf nodes, we can make sure the *BCN* is online observable. But when we used the super tree to determine the online observability of *BCNs*, we need to check the existence of loops when we build the super tree. And many nodes in the tree are repeated, these nodes will take a lot of time overhead and space overhead when we check the super tree. Therefore, we proposed the second way to determine the online observability of *BCNs* by using directed graph. By this way we can avoid checking the existence of loop and avoid checking repeated nodes. There are also other advantages which help us select the input smarter when we use the second way. All of these advantages will reduce time and space overhead to determine the initial state of a *BCN*. If a *BCN* seems to be online observable we would check it earlier by using supertree. But if a *BCN* not seems to be online observable we prefer to check it earlier by using directed graph. If we just want to find a path to determine the initial state of a *BCN* we would check it by using supertree. But if we want find all paths to determine the initial state of a *BCN* and make some optimizations we prefer to check it by using supertree.

A. Supertree

As we mentioned before, we can use the deduction function to determine the initial state of *BCNs*. According to the definition of online observability we will alternately observe the output and decide the input. When the cardinal number of the states set comes into be 1 we can determine the initial state, and stop deducing the initial state of *BCNs*. According to this process, we can define the supertree for *BCNs*. For convenience, we use the states set inside the node to represent the node, and output in the edge to represent the edge.

Definition 10 (Super Tree): The root node of the super tree is Δ_N, the leaf nodes of the super tree are the nodes with cardinal number 1 ($|S_i| = 1$). In addition to the leaf nodes, if a node S_i in the $2k+1$ ($k \in \mathbb{N}$) layer of the supertree and

$$|\texttt{Ded}\,(S_i, \varepsilon, o_j)\,| > 0,$$

then $\texttt{Ded}\,(S_i, \varepsilon, o_j)$ is one of its son nodes, and o_j is the edge from S_i to $\texttt{Ded}\,(S_i, \varepsilon, o_j)$. If a node S_i in the $2k+2$ layer of the supertree and

$$|\texttt{Ded}\,(S_i, i_p, \varepsilon)\,| = |S_i|,$$

then $\texttt{Ded}\,(S_i, i_p, \varepsilon)$ is the son node of S_i and i_p is the edge from S_i to $\texttt{Ded}\,(S_i, i_p, \varepsilon)$.

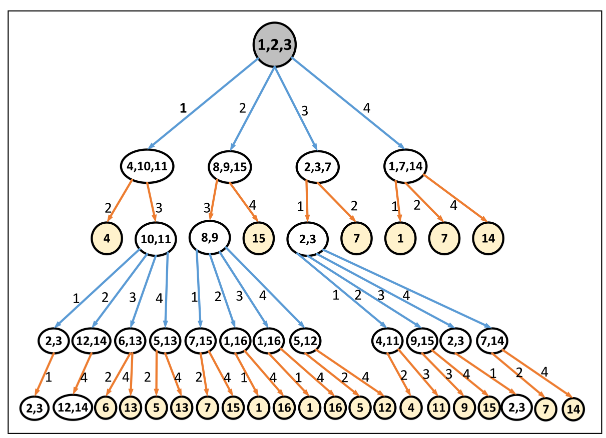

Fig. 4. Branch of the super tree which represents $\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$. The blue edges and orange edges show the observing output processes and deciding input processes, respectively. The yellow nodes are leaf nodes.

At the beginning we infer that the possible states set is Δ_N, thus the root node of the super tree is Δ_N. When the cardinal number of the possible states set turns into 1, we can determine the state of *BCN*. Therefore, the leaf nodes of the super tree are the nodes with cadinal number 1. We observe the output of *BCN* to infer the possible states set of *BCN* at first. After that, we decide the input and infer the new possible states set. We alternately observe the output and decide the input untill we can determine the state of *BCN*. Therefore we use $\texttt{Ded}\,(S_i, \varepsilon, o_j)$ to find son nodes for every

S_i in $2k+1$ layer, and using $\texttt{Ded}(S_i, i_p, \varepsilon)$ to find son nodes for every S_i in $2k+2$ layer. The formula $|\texttt{Ded}(S_i, \varepsilon, o_j)| > 0$ ensures the node $\texttt{Ded}(S_i, \varepsilon, o_j)$ has meaning. The formula $|\texttt{Ded}(S_i, i_p, \varepsilon)| = |S_i|$ guarantee we can determine the state of *BCN* in the end.

Example 3: For example, the *BCN* whose structure is depicted in Fig.2, and the updating rules of this *BCN* is described as truth table in Fig.3. The Fig.4 show branch of the tree which represents $\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$ and its deduction process. The nodes represent the states sets, the blue edges represent the observing output processes, and the orange edges represent the deciding input processes. This branch is not completed, because only the yellow nodes are the leaf nodes. If we want to find all of the ways to determine the initial state of *BCN*, we have to build the complete tree for *BCN*. This process takes many additional time and space overhead. Especially when there are loops in the tree, like the $\{\delta_{16}^2, \delta_{16}^3\}$ in fourth layer and the $\{\delta_{16}^2, \delta_{16}^3\}$ in fifth layer that will form a loop. In this case we can never build the complete tree, thus we need to check the existence of loops and omit it. There are also some nodes take the same states set which will also take additional overhead. For instance there are two nodes take the same states set $\{\delta_{16}^1, \delta_{16}^{16}\}$ in the fifth layer. However, it would be a lot easier if we only need to find a way to determine the initial state. For instance, when we find the leaf nodes δ_{16}^1, δ_{16}^7 and δ_{16}^{14} in third layer by breadth-first algorithm, we can make sure that the states set $\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$ is 1 step deterministic. After that, we use this conclusion to determine the initial state of *BCN*.

B. Directed graph

To improve the shortcomings of the way by using supertree, we proposed the way by derected graph wich may takes less time and space overhead. The most difference between supertree and derected graph is that supertree is built from the root node to leaf nodes. However, the derected graph is built from smaller nodes (contain less states) to larger nodes (contain more states). There is not any repeated node in the derected graph because any node only appears once in the graph. And even there are some loops in the derected graph, the loops would not influence us to build the directed graph completely.

The construction algorithm of derected graph is shown in the Algorithm.1. The algorithm to build nodes used in the Algorithm.1 is shown in the Algorithm.2.

The algorithm to build nodes used in the Algorithm.1 is shown in the Algorithm.2.

Some details in Algorithm.1 and Algorithm.2 are as follows:

- Build all nodes with k states: Firstly, we classify all states by their corresponding outputs, then we have all of states sets. The states set contains all states have the same corresponding outputs. Secondly, we compare k with the cardinal number Car of each states set S_i we built before. If k greater than Car, then we could not get k states from this states set S_i. Else we can get C_{Car}^k sets with k states from this states set. Finally, we use all of states sets to build nodes we need.

Algorithm 1 Algorithm to construct the directed graph of *BCNs*

Input: The algebraic forms of *BCN*
Output: The directed graph of *BCN*

$k = 1$ (The number of states in the nodes)
$Ob =$ false (The online observability of *BCN*)
N_i (Node)
i_p (Input)
$NodesArray$ (Nodes array)
Sis (The suitable inputs set of N_i)
buildnode(k)
$k = k + 1$
while NodesArray=buildnode(k)!=Null **do**
  NodesArray=buildnode(k)
  for each $N_i \in NodesArray$ **do**
    if $k == 2$ **then**
      Sis = Δ_M
    else
      Find Sis by other nodes
    end if
    for each $i_p \in Sis$ **do**
      Check N_i by i_p
      Build edges for N_i
    end for
    if N_i has not any edge. **then**
      $Ob = 0$
      return Null
    end if
  end for
  $k = k + 1$
end while
$Ob = 1$
return $NodesArray$

Algorithm 2 buildnode(int k)

Input: The number of states in the nodes k
Output: The nodes with k states whose corresponding outputs are the same

Build all nodes with p states
if Failed to build **then**
  return Null
else
  Classify these nodes
  Sort the states in these nodes
  Sort these nodes
  return nodes
end if

- Sort the states in these nodes and sort these nodes: For example, the nodes $\{\delta_{16}^1, \delta_{16}^2\}$, $\{\delta_{16}^1, \delta_{16}^3\}$ and $\{\delta_{16}^2, \delta_{16}^3\}$ shown in Fig.5.
- Find Sis by other nodes: The node N_i with k sorted states inside it, then we can use the first $k-1$ states, the last $k-1$ states and the first and last two states to

find the nodes we need. And then use these three nodes to find Sis for N_i. For example, we can search right inputs sets which make $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6\}$, $\{\delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ and $\{\delta_{16}^4, \delta_{16}^7\}$ k-step deterministic at first. After that, take the intersection of these sets to be the suitable inputs set of $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$.

- Check N_i (with states set S_i in it) by i_p: According to the order determined in previous steps, we check every node in order. If for one input i_p (which belongs to suitable inputs set Sis) implies $|\texttt{Ded}\,(S_i, i_p, \varepsilon)\,| < |S_i|$, we can make sure the i_p is a wrong input. Else if for each $O_j \in \Delta_Q$, $|\texttt{Ded}\,(S_i, i_p, o_j)\,| > 0$ and $\texttt{Ded}\,(S_i, i_p, o_j)$ is k-step deterministic then I_j is a right input. Therefore, we can connect the node S_i to each node $\texttt{Ded}\,(S_i, i_p, o_j)$ with directed edge. The colour of directed edges represent its corresponding input. Else if there exist $o_j \in \Delta_Q$ and we can not make sure whether $\texttt{Ded}\,(S_i, i_p, o_j)$ is k-step deterministic, we check it in the next round.

According the construction process, we have the definition of directed graph.

Definition 11 (Directed Graph): Every node S_i in the directed graph is k-step deterministic, and there are no duplicate nodes in the graph. For every distinct two $s_a, s_b \in S_i$ we have $Hs_a = Hs_b$. If $|S_i| = 1$, then there are not edge from it to other nodes, else if there are exist one edge i_p from it to one nodes then there exist z $(z \geq 1)$ edges contain i_p from it to nodes $S_1, \ldots, S_z$ that

$$|S_i| = |S_1| +, \ldots, |S_z|$$

and

$$\texttt{Ded}\,(S_i, i_p, \varepsilon) = S_1 \vee, \ldots, \vee S_z.$$

When we trying to build the directed graph for a *BCN*, we check whether the nodes with less states are k-step deterministic first and then check whether the nodes with more states are k-step deterministic. As the the nodes with less states are not k-step deterministic, the nodes with more states would not be k-step deterministic. Therefore, once we can find a node is not k-step deterministic we can know that Δ_N is not k-step deterministic and this *BCN* is not online observable. Moreover, we can use the nodes with less states that are k-step deterministic to help us check the nodes with more states. For instance, if the node S has two edges from it to two nodes S_1 and S_2, and we have S_1 and S_2 are k-step deterministic. In this case, we can make sure that the node S is k-step deterministic.

Based on the definitions of existing four kinds of observability, we can also use the directed graph to determine the existing second and fourth kinds of observability. When we trying to build bottom layer and penultimate layer of the directed graph, if there are exist some nodes in penultimate layer has no edges from it to other nodes. In this case this *BCN* is not satisfied existing second observability. When we trying to build edges for every layer, and if there exist one node whose right inputs set is not Δ_M, then this BCN is not satisfied existing fourth observability.

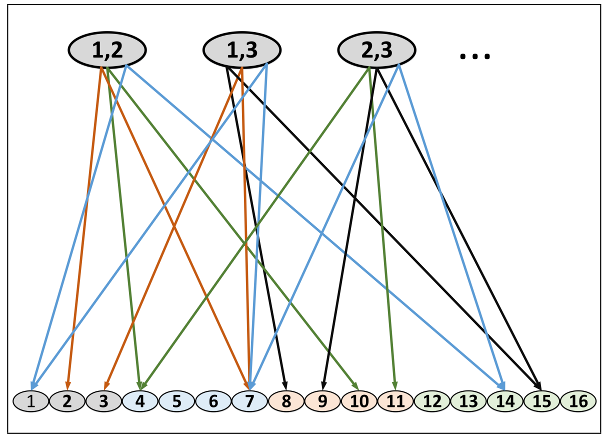

Fig. 5. Part of the directed graph which represents $\{\delta_{16}^1, \delta_{16}^2\}$, $\{\delta_{16}^1, \delta_{16}^3\}$ and $\{\delta_{16}^2, \delta_{16}^3\}$. The green, black, orange, blue edges show the inputs δ_4^1, δ_4^2, δ_4^3 and δ_4^4 respectively.

C. Complexity analysis

The way by the directed graph is better than by supertree, thus we analyze the complexity of it. We classify the states with their corresponding output. After that form the set of states set $\{S_1, S_2, \ldots, S_M\}$, and every element in a states set has the same corresponding output. That is to say, for each $S_i \in \{S_1, S_2, \ldots, S_M\}$ then for every $s_k \in S_i$ we have $Hs_k = \delta_M^i$.

Firstly, we need to calculate the upper bound of the number of the states in the directed graph nodes k we have.

$$k_{upb} = \max(|S_1|, |S_2|, \ldots, |S_M|) \tag{13}$$

The k_{upb} is the maximum value of $|S_1|, |S_2|, \ldots, |S_M|$, because the states in the directed graph nodes should have the same corresponding output.

Secondly, we need to calculate the number of nodes with k states:

$$k_{non} = C_{|S_i|}^k + \ldots + C_{|S_p|}^k \tag{14}$$

where $S_i \ldots, S_p \in \{S_1, S_2, \ldots, S_M\}$ and $|S_i|, \ldots, |S_p| \geq k$.

Thirdly we need to calculate the cardinal number of suitable inputs set of each node. Finally we need to calculate the time used to check each input is a right input for a node.

After completing the previous steps, calculate the complexity by layer by layer. But the cardinal number of suitable inputs set of a node depends on the cardinal number of it and the other three nodes used to find the suitable inputs set for it. And the time used to check whether an input is a right input for a node also depends on the updating rules of *BCNs*.

Moreover, instead of taking Δ_M as the suitable inputs set for every node in thedirected graph. We would use the other three nodes like $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6\}$, $\{\delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ and $\{\delta_{16}^4, \delta_{16}^7\}$ that are k-step deterministic to find the suitable inputs set for a node $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ with more than 2 states. By this way we can reduce the cardinal number of the suitable inputs set for every nodes with more than 2 states, and then reduce the time cost.

The reason why we can use this method is that only the input which make the subset of $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ k-step deterministic will make the $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ k-step deterministic. Furthermore, using these three nodes will be a good way to cover all the subset with cardinal number 2 of $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$. That is to say every subset s_i with cardinal number 2 included in $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ will included in $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6\}$, $\{\delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ or $\{\delta_{16}^4, \delta_{16}^7\}$. This conclusion can help us to select the nodes we need when we seek the suitable inputs set for a node. But it is hard to analyze the complexity of this method, and it makes the complexity analysis of the way by directed graph harder.

Therefore, it is hard to give a accurate complexity of the algorithm without the complete imformation of *BCNs*. We may finish this job in the furture, and we will try to use real example to do complexity analysis.

V. Applications

If the *BCNs* we research is online observable, and we have built the directed graph for them, then we can use the online observability to determine the initial states of these *BCNs*. Using the the online observability, it will need less observation costs to determine the initial state of these *BCNs*. Furthermore we can also use it to try to find the shortest path or avoid entering critical states when we determine the initial state of *BCNs*.

A. Determine the initial state

If a *BCN*is online observable and we have built the directed graph of a *BCN*, we can use it to determine the initial state of this *BCN*.

Example 4: For example, the *BCN* whose structure is depicted in Fig.2, and the updating rules of this *BCN* is described as truth table in Fig.3. The process of determining its initial state is as *Fig.6* shows.

- Firstly, we observe the output of the *BCN* mentioned before. If we observe the output is δ_4^1 then we can infer that the possible states set should be $\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$, and we record them as initial states and current states in the table.
- After that we input δ_4^1 and observe the output is δ_4^3 then we can infer that the possible states set should be $\{\delta_{16}^{10}, \delta_{16}^{11}\}$, and we record them as current states set in the corresponding position.
- Repeat the second step untill the cardinal number of the possible states set turns into 1. In that time we can determine the current state and the corresponding initial state of the *BCN*.

B. Less observation costs

Some biological systems can be depicted by *BCNs*, such as the immune systems which can be depicted as the *BCN* T-cell receptor kinetics model [12]. There exist 3 input-nodes, and 37 state-nodes in this model, therefore the model has 2^3 inputs and 2^{37} states. For the purpose of obtain the initial state of this *BCN*, we must select some state-nodes to be observe at first,

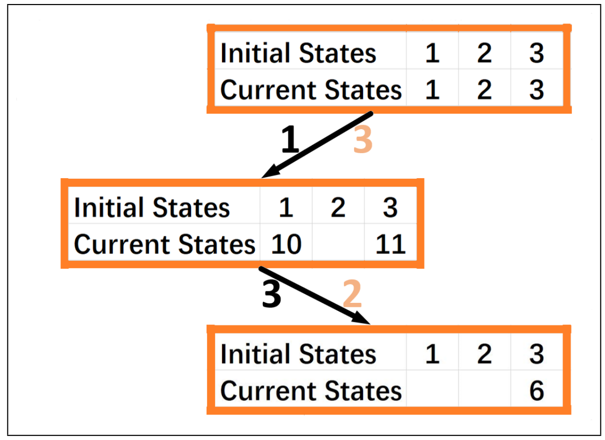

Fig. 6. The process of determing the initial state of BCNs, we change the values of current states by input and the output we observe.

and it would take many costs for us to observe the state-node values.

However if we use the online observability of *BCNs* to determine the initial state of the *BCN* T-cell receptor kinetics model, it needs less observation costs. Because compare with the existing third and fourth kind of observabilities the online observability need weakest preconditions to determine the initial state of BCNs without any presupposition. In other words, the *BCNs* needs less output-nodes to determine its initial state. As we can select less state-nodes to be observe, it will takes less ovservation costs in this example. In addition to this, there are also some other advantages, when we use the online observability of *BCNs*.

C. Finding shortest path

When we need to determine the initial state of a *BCN*, an important aspect that we will consider is to find the shortest path. In general, we can not find the shortest path definitely. Fortunately, we can use the directed graph to make the best decision. For the path, we introduce two functions $\mathtt{Pe}(S, i_p)$ and $\mathtt{Pv}(S, i_p)$ to describe its expected value and variance, respectively. For better explain our idea, we list some definitions in the following.

Some statements before definitions:

- $\{i_1, i_2, \ldots, i_z\}$: the right inputs set of S;
- $\{S_p^1, S_p^2, \ldots, S_p^k\}$: the set of state sets, and its elements corresponding to the possible outputs $\{o_1, o_2, \ldots, o_k\}$ after input i_p, for every i_p in $\{i_1, i_2, \ldots, i_z\}$.

Definition 12 ($\mathtt{Spe}(S)$):

$$\mathtt{Spe}(S) = \min(\mathtt{Pe}(S, i_1), \mathtt{Pe}(S, i_2), \ldots, \mathtt{Pe}(S, i_z)).$$

Definition 13 ($\mathtt{Pe}(S, i_p)$): When the $|S| = 1$, we have that $\mathtt{Pe}(S, i_p) = 0$ for every i_p in Δ_M. According *Definition 12*, $\mathtt{Spe}(S) = 0$ if $|S| = 1$. But when the $|S| > 1$, we have that

$$\mathtt{Pe}(S, i_p) = 1 + \frac{\sum_{j=1}^{k} \mathtt{Spe}(S_p^j)|S_p^j|}{|S|}$$

The function shortest path expected value $\mathtt{Spe}(S)$ is to find the i_p from $\{i_1, i_2, \ldots, i_z\}$ to calculat least $\mathtt{Pe}(S, i_p)$ for S.

From the definition of $\texttt{Pe}(S, i_p)$ we can know that if $|S| = 1$ then we can make sure the state of *BCNs*, and we need not input anymore to determine the state of *BCNs*. Therefore, for any input the path expected value $\texttt{Pe}(S, i_p)$ would be 0 and the shortest path expected value $\texttt{Spe}(S)$ also would be 0. But if $|S| > 1$ we still need to choose input and observe the output. Only by this way we can determine the state of of *BCNs*. Therefore, we can recursively define the $\texttt{Pe}(S, i_p)$ and $\texttt{Spe}(S)$ for each input i_p in the right inputs set. The $\texttt{Pv}(S, i_p)$ can be defined in the similar way. Hence we omit the details in this paper.

D. Avoiding entering critical states

In biological systems, some states of the genes may corresponding to unfavorable or dangerous situations [23]. So another important aspect that we will consider is to avoid entering critical states. We can also construct two functions $\texttt{Ce}(S, i_p)$ and $\texttt{Cv}(S, i_p)$ to describe expected value and variance of the times of entering critical states, the definition of $\texttt{Ce}(S, i_p)$ is as follows:

Definition 14 ($\texttt{Lce}(S)$):

$$\texttt{Lce}(S) = \min(\texttt{Ce}(S, i_1), \texttt{Ce}(S, i_2), \ldots,)\texttt{Ce}(S, i_z)$$

Definition 15 ($\texttt{Ce}(S, i_p)$): When the $|S| = 1$, and for every i_p in Δ_M, we have:

$$\texttt{Ce}(S, i_p) = |S \cap S_{cri}|$$

According *Definition 14*,

$$\texttt{Lce}(S) = \texttt{Ce}(S, i_p) = |S \cap S_{cri}|$$

But when the $|S| > 1$ we have that

$$\texttt{Ce}(S, i_p) = |S \cap S_{cri}| + \frac{\sum_{j=1}^{k} \texttt{Lce}(S_p^j)|S_p^j|}{|S|}$$

Where S_{cri} is the critical states set of the *BCN* we research. The definition of $\texttt{Ce}(S, i_p)$ has some difference with $\texttt{Pe}(S, i_p)$, because we need to use the critical states set S_{cri}. So that we can analyze the possibility of entering the critical states when we infer the possible states set of *BCNs*.

We can get the definitions of $\texttt{Cv}(S, i_p)$ in similar ways, and use all of these four functions to make the best decision we like. When we choose an input i_p with least $\texttt{Pe}(S, i_p)$, we may find the shortes path to determine the initial state. But the output of *BCNs* after input i_p is uncertain, so selecting the least $\texttt{Pe}(S, i_p)$ may leads to a very long path to determine the initial state of *BCNs*. For better performce, we the can also use the $\texttt{Pv}(S, i_p)$ to avoiding risks. The uses of $\texttt{Ce}(S, i_p)$ and $\texttt{Cv}(S, i_p)$ are similar to $\texttt{Pe}(S, i_p)$ and $\texttt{Pv}(S, i_p)$, when we trying to avoid entering critical states of *BCNs*.

In the four existing kinds of observability, we can not analyze the state of *BCNs* dynamically, so it would be hard to find the best way we like when we determine the initial state of *BCNs*. But this problem can be solved by the online observability of *BCNs*.

VI. Conclusions

In this paper, firstly we proposed the online observability of *BCNs* and define its mathematical form. Secondly we use the super tree and directed graph to determine the online observability. After introduced the ways to determine the online observability we present some applications of the online observability of *BCNs* and talk about some advantages of it.

But even we use the directed graph, it is still hard to determine the the online observability of a *BCN* with a large number of nodes. Therefore, in the future we will try to separate the *BCNs*, and then determine their online observability respectively. Furthermore, we also want to try to use some knowledge about formal methods to earn scalability for *BCNs*. In addition to the theoretical aspect, the realistic application is also very important. Hence we will also try to find some realistic example which can be modeled by *BCNs*. So that we can research these realistic examples well and determine the online observability their models for better performance.

References

[1] Waldrop and M. Mitchell, *Complexity : the emerging science at the edge of order and chaos*. Simon & Schuster, 1992.

[2] D. Cheng and H. Qi, "Controllability and observability of boolean control networks," *Automatica*, vol. 45, no. 7, pp. 1659–1667, 2009.

[3] S. A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets [j]," *Journal of Theoretical Biology*, vol. 22, no. 3, pp. 437–467, 1968.

[4] T. Akutsu, S. Miyano, and S. Kuhara, "Inferring qualitative relations in genetic networks and metabolic pathways." *Bioinformatics*, vol. 16, no. 8, pp. 727–734, 2000.

[5] I. Shmulevich, E. R. Dougherty, and W. Zhang, "From boolean to probabilistic boolean networks as models of genetic regulatory networks," *Proceedings of the IEEE*, vol. 90, no. 11, pp. 1778–1792, 2002.

[6] A. Faur, A. Naldi, C. Chaouiya, and D. Thieffry, "Dynamical analysis of a generic boolean model for the control of the mammalian cell cycle," *Bioinformatics*, vol. 22, no. 14, pp. e124–e131, 2006.

[7] D. G. Green, T. G. Leishman, and S. Sadedin, "The emergence of social consensus in boolean networks," in *Artificial Life, 2007. ALIFE '07. IEEE Symposium on*, 2007, pp. 402–408.

[8] Y. Lou and Y. Hong, "Multi-agent decision in boolean networks with private information and switching interconnection," in *Chinese Control Conference*, 2010, pp. 4530–4535.

[9] E. Fornasini and M. E. Valcher, "Observability, reconstructibility and state observers of boolean control networks," *IEEE Transactions on Automatic Control*, vol. 58, no. 6, pp. 1390–1401, 2013.

[10] T. Ideker, T. Galitski, and L. Hood, "A new approach to decoding life: systems biology," *Annu Rev Genomics Hum Genet*, vol. 2, no. 1, pp. 343–372, 2001.

[11] . Kaufman, M., . Andris, F., and . Leo, O., "A logical analysis of t cell activation and anergy," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 96, no. 7, pp. 3894–3899, 1999.

[12] S. Klamt, J. Saez-Rodriguez, J. A. Lindquist, L. Simeoni, and E. D. Gilles, "A methodology for the structural and functional analysis of signaling and regulatory networks." *Bmc Bioinformatics*, vol. 7, no. 1, p. 56, 2006.

[13] C. Biane and F. Delaplace, "Abduction based drug target discovery using boolean control network," pp. 57–73, 2017.

[14] J. Thunberg, P. Ogren, and X. Hu, "A boolean control network approach to pursuit evasion problems in polygonal environments," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 4506–4511.

[15] T. Akutsu, M. Hayashida, W. K. Ching, and M. K. Ng, "Control of boolean networks: hardness results and algorithms for tree structured networks," *Journal of Theoretical Biology*, vol. 244, no. 4, p. 670, 2007.

[16] Y. Zhao, H. Qi, and D. Cheng, "Input-state incidence matrix of boolean control networks and its applications," *Systems & Control Letters*, vol. 59, no. 12, pp. 767–774, 2010.

[17] D. Cheng, H. Qi, and Z. Li, “Identification of boolean control networks,” *Automatica*, vol. 47, no. 4, pp. 702–710, 2011.
[18] ——, *Analysis and Control of Boolean Networks*. Springer London, 2011.
[19] R. Li, M. Yang, and T. Chu, “Controllability and observability of boolean networks arising from biology.” *Chaos An Interdisciplinary Journal of Nonlinear Science*, vol. 25, no. 2, pp. 1778–1792, 2015.
[20] K. Zhang, “Observability of large-scale boolean control networks via network aggregations,” 2017.
[21] K. Zhang and L. Zhang, “Observability of boolean control networks: A unified approach based on finite automata,” *IEEE Transactions on Automatic Control*, vol. 61, no. 9, pp. 2733–2738, 2016.
[22] D. Cheng, “Semi-tensor product of matrices and its applications - a survey,” *Methods & Applications of Analysis*, vol. 321, no. 2, pp. 61–79, 2003.
[23] Z. Q. Li and J. L. Song, “Controllability of boolean control networks avoiding states set,” *Science China(Information Sciences)*, vol. 57, no. 3, pp. 1–13, 2014.