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

Abstract—**Four types of observability of Boolean control networks (*BCNs*) have been proposed to solve different problems, but all of them are offline observabilities, that we can't 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 online observability that it can determine the initial state of BCNs dynamically, by deciding input sequence and observing out sequence step by step. Compare with offline observabilities, the online obsevability has batter performance in the biological systems which can be checked at most once.**

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

I. INTRODUCTION

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*) was 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 interestes in *BNs* have been motivated by the large number of natural and artificial systems whose 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 study on control-theoretic problems of *BCNs* can date back to 2007 [11]. In which the problem of determining the controllability of *BCNs* is proved **NP**-hard in the number of nodes. And moreover 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 [4], [12], [13], [14], [2]. Furthermore controllability and observability are basic control-theoretic problems. In 2009 Cheng et. al. have developed an algebraic framework to deal with both *BNs* and *BCNs* by using *semi-tensor product* (STP) of matrices [2]. And moreover they give equivalent conditions

for controllability of *BCNs* and observability of controllable *BCNs*. To date, there are four types of observability have been proposed.

1) The first type of observability, proposed in 2009 [2], means that every initial state can be determined by an input sequence.
2) A second observability, proposed in 2010 [12], stands for that for every two distinct initial states, there exists an input sequence which can distinguish them, and this observability is determined in [15].
3) A third observability is proposed in 2011 [13], stating that there is an input sequence that determines the initial state.
4) A fourth observability is determined in 2013 [4], which is essentially the observability of linear control systems, i.e., every sufficient long input sequence can determine the initial state.

An input is the values of all input nodes on a time step, several inputs make up the input sequence. We will give the formal definition of four observables in the following pages. And all the four types of observability of *BCNs* are offline observabilities that they can't adjust the input sequence by observing the output sequence in the process of determining the initial state of *BCNs*. So we proposed the online observability that we can determine the initial state of *BCNs* dynamically. The online observability decides input sequence and observes out sequence step by step.

Contribution: To this end, firstly we give the formal definition of online observability of *BCNs*. Secondly, we raise two algorithms to determine the online observability for *BCNs*, which stem from the formal definition of online observability. Finally, we introduce some applications of the online observability of *BCNs*, and the advantages of online observability compares 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 existed four kinds of observability of *BCNs*. *Section III* represents the definition of deduce function, K steps deterministic and online observability of *BCNs*. *Section IV* represents how to determine the online observability of *BCNs* by super tree and directed graph. *Section V* introduces some applications of the online observability

of *BCNs*, and compare it with offline observabilities. *Section VI* end up with the introduction of some future works.

II. PRELIMINARIES

In this section we introduce the definition of *BCNs* and its algebraic form, then the existed four kinds of observability of *BCNs*.

A. Boolean Control Networks

A Boolean control network can be described as a directed graph, and the logical equations to describe the updating rules of the nodes in the network 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,.... For one directed edge from node v_i to node v_j means that the logic value of v_j at time step $t + 1$ is affected by the value of v_i at time step t or $t + 1$.

Note that one can only know whether or not 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*.
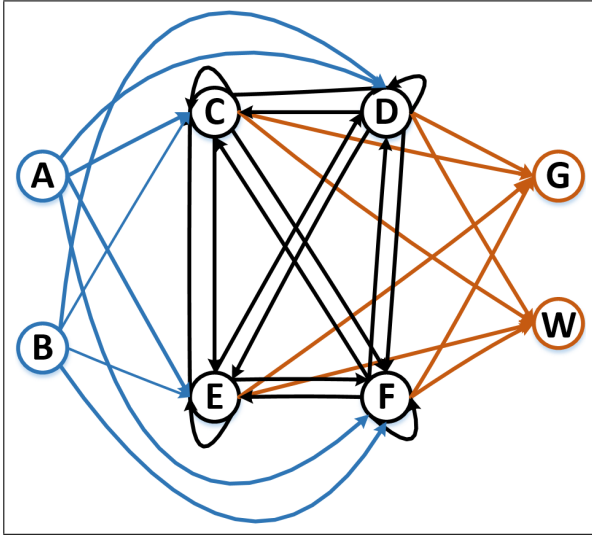

Fig. 1. A Boolean control network with two input nodes A and B, four state nodes C, D, E and F, and two output nodes G, W. We use blue, black and orange, to distinguish three kinds of nodes and three kinds of edges.

Firstly, we give a simple example to describe *BCN*.

Example 1: In *Fig.1* 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 updating rules of the *BCN* shown in the edges of *Fig.1*, and they are also described as truth table (*Fig.2*).

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

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	1	1	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	0	1	1	0	0	0	1	1	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	1	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. 2. The truth table which describe the updating rules of the *BCN* shown in *Fig.1*.

B. The Algebraic Forms of BCNs

In this paper, we investigate the following *BCN* with n state nodes, m input nodes and q output nodes. The updating rules of the *BCN* can be described as following formulas:

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

where D : the set $\{0, 1\}$; $t = 0, 1, ...$ represents the discrete time; $s \in D^n$ are state nodes; $i \in D^m$ are input nodes; $o \in D^q$ are output nodes; $f : D^{n+m} \mapsto D^n$ and $h : D^n \mapsto D^q$ are logical functions that represent the updating rules of *BCNs*. So in the previously mentioned example, we have $C(t), D(t), E(t), F(t) \in s(t)$; $A(t), B(t) \in i(t)$ 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.2*).

We use the *STP* of matrices to represent the algebraic forms of *BCNs*, the definition of *STP* is as follows.

Definition 2 (STP): [14] 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.

Since *STP* keeps most properties of the conventional product [14], the associative law, the distributive law, etc., we usually omit the symbol "$\ltimes$" hereinafter. Then we introduce some related notations before introducing the algebraic forms of *BCNs*. δ_n^i: the i-th column of the identity matrix I_n; Δ_n: the set $\{\delta_n^1, ..., \delta_n^n\}$; $\delta_n[i_1, ..., i_s]$: the logical matrix $\left[\delta_n^{i_1}, ..., \delta_n^{i_s}\right]$ $(i_1, ..., i_s \in \{1, 2, ..., n\})$; $L_{n \times s}$: the set of $n \times s$ logical matrices. [16]

So **(1)** can be quivalently represented in the following algebraic form:

$$s(t + 1) = Li(t)s(t)$$
$$o(t) = Hs(t) \tag{2}$$

where $s \in \Delta_N$, $i \in \Delta_M$, and $o \in \Delta_Q$ denote the states, inputs and outputs, respectively; $L \in L_{N \times (NM)}$; $H \in L_{Q \times N}$; that $N := 2^n$, $M := 2^m$, and $Q := 2^q$.

To construct *(2)* we give each logical value a vector form as: $T = 1 \sim \delta_2^1$, $F = 0 \sim \delta_2^2$, then the logical variable $A(t)$ takes value from thiese two vectors, i.e, $A(t) \in D := \{\delta_2^1, \delta_2^2\}$. Then $i(t) = i_1(t)...i_m(t)$; $s(t) = s_1(t)...s_n(t)$; $o(t) = o_1(t)...o_q(t)$, According to [17], for each logical function $f_p = f_p(i_1(t), ..., i_m(t), s_1(t), ..., s_n(t))$ there exists a structure matrix of f_p, say $L_p \in L_{2 \times NM}$, such as $s_p(t) \in D := \{\delta_2^1, \delta_2^2\}$ takes vector values, we have

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

for n state nodes, we have n logical matrices state nodes $L_1, ..., L_n$, if for every L_p we have $L_p = [\delta_2^{p_1}, ..., \delta_2^{p_{NM}}]$ and $L = [\delta_N^{R_1}, ..., \delta_N^{R_{NM}}]$, then we have $\delta_N^{R_1} = \delta_2^{1_1}...\delta_2^{n_1}$;...; $\delta_N^{R_{NM}} = \delta_2^{1_{NM}}...\delta_2^{n_{NM}}$. We can construct the logical matrix H in similar way.

Then the *BCN* whose structure is depicted in *Fig.1* and updating rules is described in *Fig.2* 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 $\alpha = \{10, 4, 11, 16, 9, 5, 1, 7, 15, 2, 3, 12, 7, 6, 8, 13, 8, 9, 15, 10, 14, 4, 3, 16, 1, 14, 12, 13, 5, 7, 2, 6, 7, 2, 3, 13, 13, 9, 5, 1, 16, 13, 6, 14, 11, 10, 4, 15, 1, 14, 7, 6, 9, 8, 11, 12, 5, 5, 13, 3, 10, 12, 16, 16\}$ and $\beta = \{1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4\}$; $t \in \mathbb{N}$; $s \in \Delta_{16}$; $i \in \Delta_4$; $o \in \Delta_4$.

C. Existed Four Kinds of Observability of BCNs

In order to introduce existed four kinds of observability of BCNs, we define the mappings: Let Δ_N, Δ_M, Δ_Q be three alphabets, for all $s_0 \in \Delta_N$ and all $p \in \mathbb{Z}_+$; $\mathbb{N}$ is the set of natural numbers [16],

$$\begin{aligned} L_{s_0}^p &: (\Delta_M)^p \mapsto (\Delta_N)^p, u_0...u_{p-1} \mapsto s_1...s_p \\ L_{s_0}^{\mathbb{N}} &: (\Delta_M)^{\mathbb{N}} \mapsto (\Delta_N)^{\mathbb{N}}, u_0 u_1... \mapsto s_1 s_2... \end{aligned} \quad (5)$$

$$\begin{aligned} (HL)_{s_0}^p &: (\Delta_M)^p \mapsto (\Delta_Q)^p, u_0...u_{p-1} \mapsto o_1...o_p \\ (HL)_{s_0}^{\mathbb{N}} &: (\Delta_M)^{\mathbb{N}} \mapsto (\Delta_Q)^{\mathbb{N}}, u_0 u_1... \mapsto o_1 o_2... \end{aligned} \quad (6)$$

For all $p \in \mathbb{Z}_+$, all $U = u_1...u_p \in (\Delta_M)^p$, and all $1 \geq i \geq j \geq |U|$, we use U[i,j] to denote the word $u_i...u_j$. Then existed four 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 $U \in (\Delta_M)^p$ for some $p \in \mathbb{Z}_+$ such that for all states $s_0 \neq s_0^i \in \Delta_N$, $Hs_0 = Hs_0^i$ implies $(HL)_{s_0}^p(U) \neq (HL)_{s_0^i}^p(U)$ [2].

Definition 4: The second kind of observability is that, *BCN* is called observable, if for any distinct states s_0, $s_0^i \in \Delta_N$, there exists an input sequence $U \in (\Delta_M)^p$ for some $p \in \mathbb{Z}_+$, such that $Hs_0 = Hs_0^i$ implies $(HL)_{s_0}^p(U) \neq (HL)_{s_0^i}^p(U)$ [12].

Definition 5: The third kind of observability is that, *BCN* is called observable, if there exists an input sequence $U \in$ $(\Delta_M)^p$ for some $p \in \mathbb{Z}_+$, such that for any distinct states s_0, $s_0^i \in \Delta_N$, $Hs_0 = Hs_0^i$ implies $(HL)_{s_0}^p(U) \neq (HL)_{s_0^i}^p(U)$ [13].

Definition 6: The fourth kind of observability is that, *BCN* is called observable, if for any distinct states s_0, $s_0^i \in \Delta_N$, for any input sequence $U \in (\Delta_M)^{\mathbb{N}}$, $Hs_0 = Hs_0^i$ implies $(HL)_{s_0}^{\mathbb{N}}(U) \neq (HL)_{s_0^i}^{\mathbb{N}}(U)$ [4].

We know that the fourth one implies the third one, second one and first one; the third one implies the second one and first one; the first one implies the second one [16]. And we may can't use the first one and second one to determine the initial state of *BCNs* without presupposing the initial state of *BCNs*, we can use the third one and fourth to determine the initial state of *BCNs*, but the requirements are very harsh. So we proposed the online observability of *BCNs* to solve this problem.

III. THE ONLINE OBSERVABILITY OF BCNS

In this paper we propose the online observability, the informal definition of it is as follows.

Definition 7: A *BCN* is called online observable, if for every initial state $s_0 \in \Delta_N$ can be determined by the deciding input sequence and observing output sequence step by step in finite steps without presuppose the initial state.

In this section we introduce the definition of deduce function and K steps deterministic which are the preparations of defining online observability, then the formal definition of online observability of *BCNs*.

A. Deduce Function

Different from existed four types, the observability we proposed can determine the initial state online that every input in the input sequence is decided step by step as we observe the output sequence. At the beginning, we can observe the output of *BCNs*, so we can infer the possible values of state nodes and treat them as possible states set S_0. Then as we can know the possible states set, we need to decide the input (i_0) that it will make sure any different possible states ($s_i, s_j \in S_0$) will not turn into the same state after affected by input ($Ls_i i_0 = Ls_j i_0$). After decided input, we can observe the new output, and then we can infer the new possible states set, the cardinality of possible states set after input will not lager than the cardinality of possible states set before input. If the cardinality of possible states set turn into be 1, we can determine the initial state of BCN. To simulate the deduction process, we define the deduce function at first:

Definition 8 (Deduce Function): The deduce function defined as $D(S, I, O)$, where $S \in 2^{\Delta_N}$ is the possible states set; $I \in \Delta_M$ represents the input; $O \in \Delta_Q$ represents the output; $S' = D(S, I, O) \in 2^{\Delta_N}$ is the possible states set after deduction. Based on deduction process, we have for any $s_i(t+1) \in D(S, I, O)$, there exists $s_i \in S$ that $s_i(t+1) = LIs_i(t)$ and $O = Hs_i(t+1)$.

And then we have:

$$D(\varnothing, I_i, O_i) = D(\varnothing, \varepsilon, O_i) = D(\varnothing, \varepsilon, \varepsilon) = \varnothing \quad (7)$$

$$D\left(S_i, \varepsilon, \varepsilon\right) = S_i \tag{8}$$

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

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

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

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

(7) represents that if the possible states set is $\varnothing$, no matter what you do you can only deduce $\varnothing$; if the possible states set is S_i and we don't input anything and don't observe the output, we can only deduce that the possible states set is S_i which shown in *(8)*; when the possible states set is Δ_N we observe that the outputs is δ_4^1 we can deduce that the possible states would be δ_{16}^1, δ_{16}^2 or δ_{16}^3 which shown in *(9)* ; and then we input δ_4^1, before observe the output we can only deduce the possible states would be δ_{16}^{10}, δ_{16}^4 or δ_{16}^{11} which shown in *(10)*; but if we observe that the output is δ_4^3, we can deduce that the possible state values can be δ_{16}^{10} or δ_{16}^{11} which shown in *(11)*; then if the set of states is $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6\}$ and the inputs is δ_4^3, before observe we can deduce that the possible state values can be δ_{16}^9 or δ_{16}^{13} shown in *(12)*, as the number of the possible state decreased, we may can't deduce the initial state any more.

B. K Steps Deterministic

After difined the deduce function, we can introduce the mathematical definition of online observability of *BCNs*. It may easier to be difined by programming language recursively, but we can also define its mathematical form. Before define the observability of *BCNs*, we need to difine the K $(K \geq 0)$ *steps deterministic* of the states set of *BCNs*:

Definition 9 (K Steps Deterministic): When $K = 0$, If for a set of states S_i and $|S_i| = 1$, then S_i is 0 step deterministic. It means that we can determine the state without any input and observing output if we have the cardinality of possible states set is 1. When $K > 0$, If for a set of states S_i and $|S_i| > 1$, $\exists I_i \in \Delta_M$ implies $|D\left(S_i, I_i, \varepsilon\right)| = |S_i|$, and implies "$\forall O_i \in \Delta_Q$, $|D\left(S_i, I_i, O_i\right)| \neq 0$ implies $D\left(S_i, I_i, O_i\right)$ is K_i $(K_i < K)$ stepes deterministic", then S_i is K steps deterministic.

From the definition of deterministic, if S_i is K_1 steps deterministic and $K_1 \leq K_2$, then S_i is K_2 steps deterministic. But if S_i is K_1 steps deterministic and $K_1 \geq K_2$, we can not make sure whether S_i is K_2 steps deterministic or not. So you can consider the mean of "S_i is K_i steps deterministic" as "whether we can determine the state of a *BCN* with possible states set S_i by deciding input sequence and observing out sequence step by step in K_i steps".

C. Online Observability

The formal definition of the online observability of *BCNs* is as follows.

Definition 10 (Online Observability of BCNs): If $\forall O_i \in \Delta_Q$ and $|D\left(\Delta_N, \varepsilon, O_i\right)| \neq 0$, $\exists K_i \geq 0$ implies $D\left(\Delta_N, \varepsilon, O_i\right)$

is K_i stepes deterministic, then this *BCN* is online observable. Even simpler, if $\exists K_i \geq 0$ implies Δ_N is K_i stepes deterministic, then this *BCN* is online observable. The difference is that whether we observe the corresponding output of the initial state of *BCN* at first, for better performance the former definition would be better.

Because in the existed second kind of observability, we presuppose the initial state of *BCNs*, 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 other kinds of initial states turn into be the same state, so that other kinds of initial states can't be distinguished anymore. This problem has to be considered in the online obervability of *BCNs*, so the online observability implies the existed first kind of observability, and then implies the existed second kind of observability. In the existed third kind of observability, there has to exist an input sequence that can distinguish any distinct states, but in online observability we can use different input sequences to distinguish different states set which classified by their corresponding output. So we have the existed third kind of observability implies the online observability of *BCNs*, then the existed fourth kind of observability implies the online observability.

When I learn the existed four kinds of observability of *BCNs*, I find that if we want determine the initial state of a *BCNs* 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, we can determine it, but if not, we need to guess again and input the corresponding input sequence untill we determine the initial state of the *BCN*. But if we can't repeat this process, we may can't determine the initial state of the *BCN* any more. 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 can't we try to find corresponding input sequence for them? 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* without presuppose the initial state is the online observability of *BCNs*. By this definition we can build *BCNs* with least output nodes when we want to determine the initial state of *BCNs*.

IV. DETERMINING THE ONLINE OBSERVABILITY OF BCNS

In this paper we proposed two ways to determine the online observability of *BCNs*, the first one is by supertree; the second one is by directed graph. The construction process of supertree simulates deduction process mentioned before. Then we check the tree based on the definition of online observability of *BCNs* depth first or breadth first. When we used the super tree to determine the observability of *BCNs*,

we need to check the existence of loops, and many nodes in the tree are repeated, which will take a lot of time overhead and space overhead. So we proposed the second way which determine the online observability by directed graph, in this way we can avoid checking the existence of loop and avoid checking repeated nodes. There are also other advantages like determining observability earlier and select the input smarter, which will reduce time and space overhead.

A. Supertree

As we mentioned before when we want to determine the initial state of *BCNs*, we can use the deduce function. Then according to the definition of online observability we will alternately observing the output and deciding 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 11 (Supertree): The root node of the supertree should be Δ_N, if ($|S_i| = 1$) then S_i would be the leave node, else if a node S_i in the $2k+1$ ($k \geq 0$) layer of the supertree, then its son nodes would be $D(S_i, \varepsilon, O_i)$, O_i would be the edge from S_i to $D(S_i, \varepsilon, O_i)$ where $|D(S_i, \varepsilon, O_i)| > 0$, if a node S_i in the $2k+2$ ($k \geq 0$) layer of the supertree, then its son nodes would be $D(S_i, I_i, \varepsilon)$ where $|D(S_i, \varepsilon, O_i)| > 0$, I_i would be the edge from S_i to $D(S_i, I_i, \varepsilon)$.
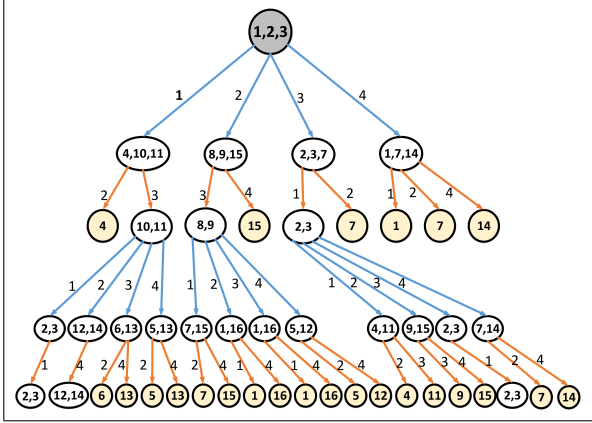

Fig. 3. Branch of the tree which represents $\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$. The blue edges and orange edges show the observing output processes and deciding input processes respectively; the yellow nodes are leaf nodes.

For example, the *Fig.3* show branch of the tree which represents $\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$, the nodes represent the states sets, the blue edges represent the observing output processes, and the orange edges represent the deciding input processes. And only the yellow nodes are the leave nodes, so you can see that this branch is not completed. If we want to find all of the ways to determine the initial state, we have to build the complete tree, it will 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 you can never build the complete tree, so you 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, like that there two nodes which take $\{\delta_{16}^1, \delta_{16}^{16}\}$ in the fifth layer. But if we only need to find a ways to determine the initial state, when we find the leaf nodes δ_{16}^1, δ_{16}^7 and δ_{16}^{14} in third layer by breadth-first algorithm, you can make sure that the states set $\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$ is 1 step deterministic.

B. Directed Graph

To improve the shortcomings of the way by supertree, we proposed derected graph wich will 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, but the derected graph is built from smaller nodes (contain less states) to larger nodes (contain more states). The construction algorithm of derected graph is showed in the *Algorithm 1*, the algorithm to build nodes used in the *Algorithm 1* is showed in the *Algorithm 2*.

And the way to check whether this input can make the nodes z ($z \geq 0$) steps deterministic: According to the order determined in previous steps, we check every node one by one. If for one input I_i which belongs to suitable inputs set of the states set S_i implies $|D(S_i, I_i, \varepsilon)| < |S_i|$, we can make sure the I_i is a wrong input; else if for all $O_i \in \Delta_Q$ and $|D(S_i, I_i, O_i)| > 0$, the $D(S_i, I_i, O_i)$ is Z ($Z \geq 0$) steps deterministic then I_i is a right input, then connect the node S_i to all nodes $D(S_i, I_i, O_i)$ with directed edges, the colour of directed edges represent the corresponding input; else if there exist $O_i \in \Delta_Q$ and we can not make sure whether $D(S_i, I_i, O_i)$ is Z steps deterministic, we check it in the next round.

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

Definition 12 (Directed Graph): For every node S_i in the directed graph, $\exists K \geq 0$ implies S_i is K stepes deterministic; for every distinct two $s_a, s_b \in S_i$ we have $Hs_a = Hs_b$; if $|S_i| = 1$ there are not edge from it to other nodes, else if there are exist one edge contains I_i from it to other nodes then there exist p ($p \geq 1$) edges contain I_i from it to nodes $S_1, ..., S_p$ that $|S_i| = |S_1| +, ..., |S_p|$ and $D(S_i, I_i, \varepsilon) = S_1 \vee, ..., \vee S_p$.

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

C. Complexity Analysis

The way use the directed graph to determine the online observability of *BCNs* is better than by supertree, so we analyze the complexity of it. We classify the states with their

Algorithm 1 Algorithm to construct the directed graph of BCNs

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

1: Define a int variable $k = 0$, to represent the number of states in the nodes.
2: Define a boolean type variable $Ob = 0$, to represent the online observability of BCN.
3: **while** buildnode(k)==1 **do**
4:    $k = k + 1$
5:    **if** $k == 2$ **then**
6:       We make Δ_M as the suitable inputs set for every node with k states.
7:    **else**
8:       **if** $k > 2$ **then**
9:          We use two nodes with $(k - 1)$ states and a node with two states to find the suitable inputs set for every node with k states. (For example, we can search inputs sets which make $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6\}$, $\{\delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ and $\{\delta_{16}^4, \delta_{16}^7\}$ z ($z \geq 0$) steps deterministic, and then take the intersection of these sets to be the suitable inputs set of $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$.)
10:       **end if**
11:    **end if**
12:    **for** every node with k states **do**
13:       **for** every input in the suitable input set of the nodes **do**
14:          Check whether this input can make the nodes z ($z \geq 0$) steps deterministic, and build edges for this node.
15:       **end for**
16:    **end for**
17:    **if** there exist one node without any edge connect it with other nodes **then**
18:       $Ob = 0$ return Ob
19:    **end if**
20: **end while**
21: $Ob = 1$ return Ob

Algorithm 2 Algorithm to build nodes with k states

Input: The number of states in the nodes p
Output: The nodes with p states

1: int buildnode(int p)
2: {
3: $p = p + 1$
4: Try to build all the nodes with p states whose corresponding outputs are the same, and classify them by their corresponding outputs.
5: **if** Failed to build **then**
6:    return 0;
7: **else**
8:    Sort the states inside the nodes from small to large, and then sort the nodes based on the values inside the 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.4.*)
9:    return 1;
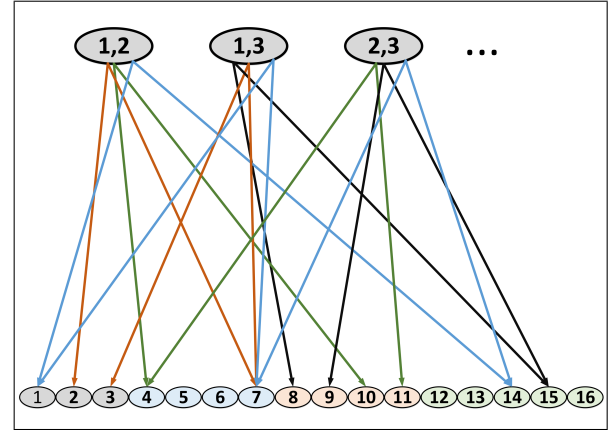10: **end if**
11: }

Fig. 4. 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.

corresponding output and form the set of states with the same corresponding output: $S_1, S_2,...,S_M$.

Firstly, we need to calculate the upper bound of the number of the states in a directed graph nodes k, then :

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

Secondly, the number of each nodes with k states:

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

where $|S_i|, ..., |S_p| \geq k$.

And then the the cardinal number of suitable inputs set of each node, and the time used to check each input is a right input for a node. Finally, calculate the complexity by layer by layer. But the cardinal number of suitable inputs set of a node depends on the number of the states in it, and the other three

nodes which used to find the suitable inputs set for it. And the time used to check an input is a right input for a node of directed graph also depends on the update rules of the *BCNs*. So it is hard to give a accurate complexity of the algorithm without the complete imformation of *BCNs*.

We can use other three nodes like $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6\}$, $\{\delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ and $\{\delta_{16}^4, \delta_{16}^7\}$ which is z ($z \geq 0$) steps deterministic to find suitable inputs set for $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$. Because only the input which make the subset of $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ z steps deterministic will make the $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ z steps deterministic, and use the this three nodes will be a convenient way to cover all the subset of $\{\delta_{16}^4, \delta_{16}^5, \delta_{16}^6, \delta_{16}^7\}$ which with cardinal number 2. By this way we can reduce the cardinal number of suitable inputs set for every nodes with more than 2 states, then reduce the time cost.

V. APPLICATIONS

If the *BCN* we research is online observable, and we have built the directed graph for it, then we can use it to determine the initial state of *BCN*. And we can also use it to try to find the shortest path or avoid entering critical states when we determine the initial state of *BCN*. Because the output we observe is undeterminable, we use expected value and variance of the length of path and the times of entering critical states to help us to chose the input.

A. Determine the Initial State

After we build the directed graph of the *BCN*, we can use it to determine the initial state of *BCNs* as *Fig.5* shows. First, we observe the output of the *BCN* mentioned before, if we observe δ_4^1 that we can infer that the possible states set should be $\{\delta_{16}^1, \delta_{16}^2, \delta_{16}^3\}$, record them as initial states and current states in the table. And then, we input δ_4^1 and observe δ_4^3, we can infer that the possible states set can be $\{\delta_{16}^{10}, \delta_{16}^{11}\}$, record them as current states set in the corresponding position. Input and output again and again untill we can infer only one possible state, in that time we can determine the current state and the corresponding initial state of the *BCN*.
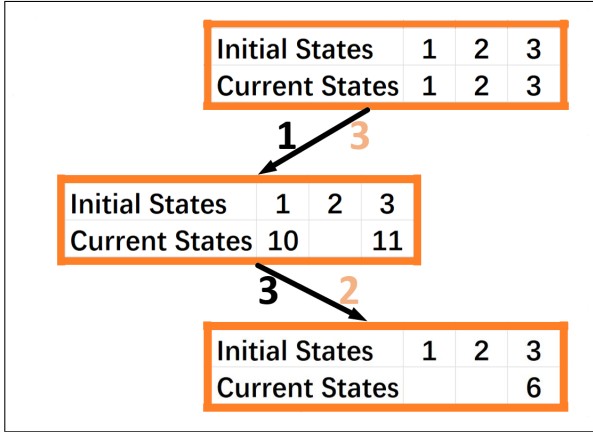

Initial States	1	2	3
Current States	1	2	3

1   3

Initial States	1	2	3
Current States	10		11

3   2

Initial States	1	2	3
Current States			6

Fig. 5. The process of determing the initial state of BCNs, we change the values of current states by input and the output we observe.

B. Less Observation Costs

Some biological systems, such as the immune systems which can be depicted as the *BCN* T-cell receptor kinetics model [18]. There exist 3 input nodes, and 37 states nodes in this model, therefore the model has 2^3 inputs and 2^{37} states. For the purpose of obtain the initial state of the *BCN*, one must select some state nodes to be observe at first.

And if we use the online observability of *BCNs* to determine the initial state of the *BCN* T-cell receptor kinetics model, it will needs less observation costs. Compares with the existed third and fourth kind of observabilities which can also determine the initial state, the online observability need weaker preconditions. Then the online observability needs less output nodes to be observe and less observation costs. In addition to this, there are also some other advantages, when we use the online observability of *BCNs*.

C. Find 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. Maybe we can't find the shortest path definitely, but we can use the directed graph to make the best decision. We introduce two functions $SPE(S_i, I_i)$ and $SPV(S_i, I_i)$ to describe the shortest path expected value and shortest path variance, S_i is the possible states set and I_i is the input we chose, the definition of $SPE(S_i, I_i)$ is as follows:

When $|S_i| = 1$:

For all $I_i \in \Delta_M$, we have $SPE(S_i, I_i) = 0$, then the least shortest path expected value $LLCE(S_i) = 0$.

When $|S_i| > 1$:

$\{I_1, I_2, ...\}$: the right inputs set of S_i; $\forall I_i \in \{I_1, I_2, ...\}$, and $\{S_i^1, S_i^2, ...\}$ is corresponding to all possible output O_i after input I_i; $SPE(S_i, I_i) = 1 + ((LSPE(S_i^1)|S_i^1| + LSPE(S_i^2)|S_i^2| + ...)/|S_i|)$; $LSPE(S_i) = Min(SPE(S_i, I_1), SPE(S_i, I_2), ...)$.

D. Avoid Entering Critical States

In biological systems, some states of the genes may corresponding to unfavorable or dangerous situations [19]. So another important aspect that we will consider is to avoid entering critical states. We can also construct two functions $LCE(S_i, I_i)$ and $LCV(S_i, I_i)$ to describe least expected value and variance of the times of entering critical states, the definition of $LCE(S_i, I_i)$ is as follows:

CS: the critical states set of the *BCN* we research.

When $|S_i| = 1$:

For all $I_i \in \Delta_M$, we have $LCE(S_i, I_i) = |S_i \cap CS|$, then the least shortest path expected value $LLCE(S_i) = |S_i \cap CS|$.

When $|S_i| > 1$:

$\{I_1, I_2, ...\}$: the right inputs set of S_i; $\forall I_i \in \{I_1, I_2, ...\}$, and $\{S_i^1, S_i^2, ...\}$ is corresponding to all possible output O_i after input I_i; $LCE(S_i, I_i) = (|S_i \cap CS| + (LLCE(S_i^1)|S_i^1| + LLCE(S_i^2)|S_i^2| + ...)/|S_i|)$; $LLCE(S_i) = Min(LCE(S_i, I_1), LCE(S_i, I_2), ...)$.

Then we can get the definitions of $SPV(S_i, I_i)$ and $LCV(S_i, I_i)$ in similar ways, and use them to make the best decision we like. When we choos an input with least $SPE(S_i, I_i)$, we may find the shortes path to determine the initial state. But the output of *BCNs* is uncertain, so selected the least $SPE(S_i, I_i)$ may leads to a very long path. For better performce, we the can also use the $SPV(S_i, I_i)$ to avoiding risks. The uses of $LCE(S_i, I_i)$ and $LCV(S_i, I_i)$ are similar to $SPE(S_i, I_i)$ and $SPV(S_i, I_i)$, when we trying to avoid entering critical states of *BCNs*.

In the existed four kinds of observability, we can not analyze the state of *BCNs* dynamically, maybe it would be a little hard to find the best way we like to determine the initial state of *BCNs*.

VI. CONCLUSIONS

In this paper, we proposed online observability of *BCNs* and define its mathematical form; then we use the super tree and directed graph to determine the online observability; after determined the online observability we use it to try to find the shortest path and avoid entering critical states when we determining the initial state of *BCNs*.

But even we use the directed graph, it is still hard to determine the the online observability of *BCNs* which with a large number of nodes. So, in the future we will try to separate the *BCNs*, and then determine their online observability; and try to use some knowledge about formal methods for better scalable of *BCNs*. In addition to the theoretical aspects, the realistic application is also very important, we will also try to find some realistic example which can be modeled by *BCNs*, then determine their online observability 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] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[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] 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.

[12] 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.

[13] D. Cheng, H. Qi, and Z. Li, "Identification of boolean control networks," *Automatica*, vol. 47, no. 4, pp. 702–710, 2011.

[14] ——, *Analysis and Control of Boolean Networks*. Springer London, 2011.

[15] 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.

[16] 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.

[17] D. Cheng, "Semi-tensor product of matrices and its applications - a survey," *Methods & Applications of Analysis*, vol. 321, no. 2, pp. 61–79, 2003.

[18] 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.

[19] 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.