KYBERNETIKA — VOLUME *54* (2018), NUMBER 5, PAGES 1091–1104

RECONSTRUCTIBILITY OF BOOLEAN CONTROL NETWORKS WITH TIME DELAYS IN STATES

Ping Sun, Lijun Zhang and Kuize Zhang

This paper deals with the reconstructibility of Boolean control networks (BCNs) with time delays in states. First, a survey on the semi-tensor product, weighted pair graph, constructed forest and finite automata is given. Second, by using the weighted pair graph, constructed forest and finite automata, an algorithm is designed to judge whether a Boolean control network with time delays in states is reconstructable or not under a mild assumption. Third, an algorithm is proposed to determine the current state. Finally, an illustrative example is given to show the effectiveness of the proposed method.

Keywords: Boolean control network, reconstructibility, semi-tensor product of matrices, weighted pair graph, finite automaton, formal language

Classification: 94C10, 03D05, 68Q45, 05C22

1. INTRODUCTION

In the investigation of cellular networks, Kauffman firstly proposed the Boolean network [16]. It has become a powerful tool in describing, analyzing and simulating the cellular networks and genetic regulatory networks [9, 21]. In Boolean networks, a gene expression level is modeled by a binary value, 0 or 1, indicating two transcriptional states, either active (on) or inactive (off), respectively. Moreover, interactions between the state of each gene can be determined by Boolean functions, which calculate the state of a gene from the activation or inhibition of other genes. Boolean networks have been investigated widely. It was pointed out in [1] that "One of the major goals of systems biology is to develop a control theory for complex biological systems." Hence, it is of practical significance to investigate the control problems of Boolean networks.

As for Boolean control networks, it was pointed out by [14] that "Gene-regulatory networks are defined by trans and cis logic. ... Both of these types of regulatory networks have input and output." From here one easily sees that a Boolean network with input(s) and output(s), called a Boolean control network, is a proper way to describe the dynamics of gene-regulatory networks.

Recently, a new tool, called the semi-tensor product (STP) of matrices [3], has been used to study Boolean (control) networks, including the analysis of Boolean networks [4, 5, 29] and the control design of Boolean control networks [6, 7]. In [5], the algebraic

DOI: 10.14736/kyb-2018-5-1091

form of a Boolean network is introduced, which provides a framework for this new approach. Using it, some formulas are obtained to calculate the fixed points, cycles, basins of the attractors and transient periods. In [29], using the input-state analysis, the structure of attractors of a Boolean network is investigated. The basic idea is to convert the logical dynamic model of a Boolean (control) network into a discrete time dynamic system by using the vector expression of logic. The series of works showed that the semi-tensor product is a powerful tool in analyzing the structure of Boolean networks and the synthesis of Boolean control networks. On the other hand, the STP method has wide applications in the engineering related fields [12] and in logical networks and other finite-valued systems [20].

Systematic analysis of biological systems is an important topic in systems biology. Reconstructibility is one of the fundamental concepts in control theory of Boolean control networks. The concept of reconstructibility of Boolean control networks was proposed in [10, 28]. There have been many studies on the reconstructibility of dynamic systems, e. g. [2, 22, 23]. But when it comes to the reconstructibility problem of Boolean (control) networks, there have been only very few results, because there are a shortage of systematic tools to deal with logical systems. There are some recent papers concerning this problem, e. g. [10, 28]. They all study the reconstructibility of Boolean (control) networks without time delays. But we can see that time delay phenomena are very common in real world, for instance, economic, biological and physiological systems and so on. It is well known that, in many cases, time delay cannot be avoided in practice and it often results in some poor performance. Also, the presence of time delays makes the analysis of the reconstructibility of Boolean (control) networks much more complicated. There has been some literature that discussed Boolean (control) networks with time delays in states. In [18, 19, 13], the controllability and observability of Boolean control networks with time-invariant delays in states were investigated, respectively. In [26, 27], the controllability of time-variant Boolean networks and Boolean control networks with multiple time-variant and bounded time delays in states by showing simple test matrix criteria were investigated, respectively. But to the best of our knowledge, there has been no result in the literature concerning the reconstructibility of Boolean control networks with time delays in states. Motivated by the above analysis, in our paper, we investigate the reconstructibility problem of Boolean control networks with time-variant delays in states.

The rest of this paper is organized as follows. Section 2 introduces necessary preliminaries about the semi-tensor product of matrices, Boolean control networks with their algebraic forms and some notations. Section 3, first, we convert Boolean control networks with time-variant delays into discrete time delayed systems using the semi-tensor product of matrices. Second, we show how to construct a weighted pair graph for a Boolean control network with time-variant delays. Third, algorithms are designed to use the weighted pair graph of a Boolean control network to determine its reconstructibility. Lastly, algorithms are designed to determine the current state of the reconstructible Boolean control networks. In section 4, the efficiency of the results is demonstrated using examples. The last section is a short conclusion.

2. PRELIMINARIES

In this paper, the matrix product used is the semi-tensor product (STP) of martrices. Consider an $m \times n$ martrix A, a $p \times q$ martrix B. The STP of A and B, denoted by $A \ltimes B$, is defined as $A \ltimes B := (A \otimes I_{\frac{t}{n}})(B \otimes I_{\frac{t}{p}})$, where t is the least common multiple of n and p, $\otimes$ denotes the Kronecker product. The STP of martrices is a generalization of the conventional matrix product. Based on this, we can omit the symbol $\ltimes$. [8] introduces the definition and the basic properties of STP in detail.

Next, we define some notations:

- $\mathbb{Z}$: the set of integers
- $\mathbb{N}$: the set of all natural numbers
- $\mathcal{D}$: the set $\{0, 1\}$
- δ_n^i: the ith column of the identity matrix I_n
- Δ_n: the set $\{\delta_n^1, \ldots, \delta_n^n\}$ $(\Delta := \Delta_2)$
- $\delta_n[i_1, \ldots, i_s]$: the logical matrix $[\delta_n^{i_1}, \ldots, \delta_n^{i_s}]$ $(1 \leq i_1, \ldots, i_s \leq n$ and $i_1 \ldots, i_s$ are integers)
- $\mathcal{L}_{n \times s}$: the set of all $n \times s$ logical matrices, i. e., $\{\delta_n[i_1, \ldots, i_s] | i_1, \ldots, i_s \in \{1, 2, \ldots, n\}\}$
- $[1, N]$: the consecutive integer set $\{1, 2, \ldots, N\}$
- $\mathbf{1}_k$: $\sum_{i=1}^{k} \delta_k^i$
- $Col_i(A)$: The ith column of a matrix A.

3. MAIN RESULTS

In this paper, we mainly investigate the following BCNs with n state nodes, m input nodes and q output nodes, and time delays in states:

$$\begin{cases} x_1(t+1) = f_1(u_1(t), \ldots, u_m(t), \\ \qquad\qquad x_1(t-\tau(t)), \ldots, x_n(t-\tau(t))), \\ x_2(t+1) = f_2(u_1(t), \ldots, u_m(t), \\ \qquad\qquad x_1(t-\tau(t)), \ldots, x_n(t-\tau(t))), \\ \qquad\qquad \vdots \\ x_n(t+1) = f_n(u_1(t), \ldots, u_m(t), \\ \qquad\qquad x_1(t-\tau(t)), \ldots, x_n(t-\tau(t))), \\ y_j(t) = h_j(x_1(t), \ldots, x_n(t)), \quad j = 1, \ldots, q, \end{cases} \tag{1}$$

where $t = t_0, t_0 + 1, \cdots \in \mathbb{Z}$ denote time steps; $x_i(t), u_k(t), y_j(t) \in \mathcal{D}$, $i = 1, 2, \ldots, n$, $k = 1, 2, \ldots, m$, $j = 1, 2, \ldots, q$, $f_1, \ldots, f_n : \mathcal{D}^{n+m} \to \mathcal{D}$, $h_1, \ldots, h_q : D^n \to D$ are

Boolean functions. $\tau : \{t \in \mathbb{Z} : t \geq t_0\} \to \mathbb{N}$ is a mapping, called the time delay function. Throughout this brief, without loss of generality, we assume that $t - \tau(t) \geq t_0 - \tau(t_0)$ $\forall t \geq t_0$ to ensure that the system (1) has a starting point. The trajectory of the system (1) is determined by its initial state sequence $x(t_0 - \tau(t_0)), x(t_0 - \tau(t_0) + 1), \ldots, x(t_0)$ and the control sequence.

Hereinafter, denote $N := 2^n$, $M := 2^m$ and $Q := 2^q$ for short, respectively.

By using STP [25], (1) is represented equivalently as

$$\begin{aligned} x(t+1) &= Lu(t)x(t-\tau(t)), \\ y(t) &= Hx(t), \end{aligned} \tag{2}$$

where $x(t) \in \Delta_N$, $u(t) \in \Delta_M$ and $y(t) \in \Delta_Q$ denote the state, input and output, respectively. $L \in \mathcal{L}_{N\times(NM)}$, $H \in \mathcal{L}_{Q\times N}$, t and $\tau(t)$ are the same as those in the system (1).

3.1. Reconstructibility of Boolean control networks

In the following, we consider the reconstructibility problem of BCN (1), equivalently (2).

Definition 3.1. Consider a BCN (2). For any given initial time t_0, any given time delay function $\tau(t) : \{t_0, t_0 + 1, \ldots\} \to \mathbb{N}$, a BCN (2) is called reconstructible, if there exists an input sequence $\{u(t_0), u(t_0 + 1), \ldots\} \subset \mathcal{D}^m$ (called "homing input sequence") such that there exists a time step T, the state at any time step $t \geq T$ is determined by the corresponding output sequence.

3.2. Finite automata and formal languages

We next introduce finite automata and formal languages to support our discussion.

We use Σ, a nonempty finite set to denote the alphabet Elements of Σ are called letters. A word is a finite sequence of letters. The empty word is denoted by ϵ. The set of all words over alphabet Σ is denoted by Σ^*. For example,

$$\{0, 1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}.$$

A formal language is defined as a subset of Σ^*.

A language over alphabet Σ is called regular, if it is recognized by a deterministic finite automaton (DFA).

A DFA is a 5-tuple $A = (S, \Sigma, \sigma, s_0, F)$:

- Finite state set S. At all times the internal state is some $s \in S$.
- Input alphabet Σ. The automaton only operates on words over the alphabet.
- The transition partial function describes how the automaton changes its internal state. It is a partial function

  $$\sigma : S \times \Sigma \to S$$

  from (state, input letter)-pairs to states, that is, σ is a function defined on a subset of $S \times \Sigma$. If the automaton is in state s, the present input letter is a, then the automaton changes its internal state to $\sigma(s, a)$ and moves to the next input letter, if σ is well defined at (s, a); and stop, otherwise.

- Initial state $s_0 \in S$ is the internal state of the automaton before any letters have been read.

- Set $F \subset S$ of final states specifies which states are accepting and which are rejecting. If the internal state of the automaton, after reading the whole input, is some state of F then the word is accepted, otherwise rejected.

We call a DFA complete if σ is a function from $S \times \Sigma$ to S.

In order to represent regular languages, we introduce an extended transition function $\sigma^* : S \times \Sigma^* \to S$. σ^* is recursively defined as

- $\sigma^*(s, \epsilon) = s$ for all $s \in S$.

- $\sigma^*(s, wa) = \sigma(\sigma^*(s, w), a)$ for all $s \in S$, $w \in \Sigma^*$ and $a \in \Sigma$, if σ is well defined at $(\sigma^*(s, w), a)$ and σ^* is defined on (s, w).

Particularly, one has for all $s \in S$, all $a \in \Sigma$, $\sigma^*(s, a) = \sigma(\sigma^*(s, \epsilon), a) = \sigma(s, a)$, if σ is well defined at (s, a). Hence we will use σ to denote σ^* briefly, since no confusion will occur. Hereinafter, if we write "$\sigma(s, a)$", we mean that σ is well defined at (s, a).

Given a DFA $A = (S, \Sigma, \sigma, s_0, F)$. A word $w \in \Sigma^*$ is called accepted by this DFA, if $\sigma(s_0, w) \in F$. A language $L \subset \Sigma^*$ is called recognized by this DFA, if $L = \{w \in \Sigma^* | \sigma(s_0, w) \in F\}$, and is denoted by $L(A)$.

In order to visualize and represent a DFA and transform a BCN into a DFA related to its reconstructibility, we introduce the transition graph of a DFA $A = (S, \Sigma, \sigma, s_0, F)$.

A weighted digraph $G_A = (V, E, W)$ is called the transition graph of the DFA A, if the vertex set $V = S$, the edge set $E \subset V \times V$ and the weight function $W : E \to 2^{\Sigma}$, where 2^{Σ} is the power set of Σ, are defined as follows: For all $(s_i, s_j) \in V \times V$, $(s_i, s_j) \in E$ iff there is a letter $a \in \Sigma$ such that $\sigma(s_i, a) = s_j$. If $(s_i, s_j) \in E$, then $W((s_i, s_j))$ equals the set of all letters $s \in \Sigma$ such that $\sigma(s_i, a) = s_j$.

In a transition graph of a DFA, usually an input arrow is added to the vertex denoting the initial state, double circles are used to denote the final states, the curly bracket "{}" in the weights of edges are not drawn. See the following example.

Example 3.2. The graph in Figure 1 represents the DFA $A = (\{s_0, s_1, s_2\}, \{0, 1\}, \sigma, s_0, \{s_0, s_1\})$, where

$$\begin{aligned} &\sigma(s_0, 0) = s_0, \quad \sigma(s_1, 0) = s_0, \quad \sigma(s_2, 0) = s_2, \\ &\sigma(s_0, 1) = s_1, \quad \sigma(s_1, 1) = s_2, \quad \sigma(s_2, 1) = s_1. \end{aligned}$$

It is easy to see that $\epsilon \in L(A)$, $0101111 \in L(A)$ and $010110 \notin L(A)$.

Proposition 3.3. (Zhang et al. [28]) Given a DFA $A = (S, \Sigma, s_0, F)$, assume that $F = S$, and for each s in S, there is a word $u \in \Sigma^*$ such that $\sigma(s_0, u) = s$. Then $L(A) = \Sigma^*$ iff A is complete.

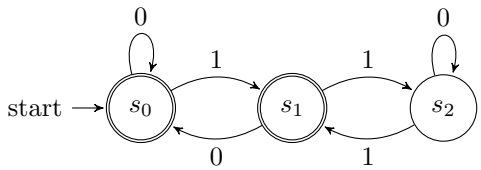

Fig. 1. The transition graph of the DFA A in Example 3.2.

3.3. Constructed forest

The concept of the "constructed forest" [25] is introduced to support our discussion.

Definition 3.4. (Zhang and Zhang [25]) A directed graph $G(V, E)$ is said to be the constructed forest of (2) if the vertex set $V = \{t \in \mathbb{Z} : t \geq t_0 - \tau(t_0)\}$, i. e., the time sequence of (2), and the edge set $E = \{(t', t'') : t' = t'' - 1 - \tau(t'' - 1)\} \subset V \times V$.

For the sake of discussion, we present several concepts as follows

1. Let $\{T_{t_0-\tau}, T_{t_0-\tau+1}, \ldots, T_{t_0}\}$ be the constructed forest $G(V, E)$, where each T_i is a tree.

2. Let $P_{t_0-\tau+i}$ and N_i be any one given longest path of the tree $T_{t_0-\tau+i}$ and the length of $P_{t_0-\tau+i}$, respectively, $i = 0, 1, \ldots, \tau$ (If $P_{t_0-\tau+j}$ has an infinite number of vertices, set $N_j = +\infty, j = 0, 1, \ldots, \tau$.).

3. Let P_o and N_o be any one given shortest path of the set $\{P_{t_0-\tau}, P_{t_0-\tau+1}, \ldots, P_{t_0}\}$ and the length of P_o, respectively. P_o is called an observability contructed path.

To obtain the main result, we show that any given observability constructed path P_o can be identified with a subsystem that generates it.

Denote by

$$\{\mathbf{t}_0, t_1, \ldots, t_N\}(or\{\mathbf{t}_0, t_1, \ldots\}) \subset \mathbb{Z} \tag{3}$$

one of the observability constructed paths of (2), where $t_0 - \tau(t_0) \leq \mathbf{t}_0 \leq t_0$, $t_{i+1} > t_i$ for all $i \geq 0$. Then, we identify the observability constructed path (3) with the following subsystem (4) of (2). In (4), we still use t_0 to denote $\mathbf{t}_0$ when confusion does not occur

$$\begin{cases} x(t_{k+1}) = Lu(t_{k+1} - 1)x(t_k) \\ y(t_k) = Hx(t_k) \end{cases} \tag{4}$$

where $x \in \Delta_{2^n}$, $u \in \Delta_{2^m}$, L, H are the same as those in (2). In fact, it is easy to see that any path of (2) with its root in the set $\{t_0 - \tau, t_0 - \tau + 1, \ldots, t_0\}$ has the same form as (4).

Note that (4) is a system with no time delays if the subscript of t is regarded as its time sequence.

Next we give an assumption for the system (2), and an equivalent condition for the reconstructibility of (2) under this assumption.

Assumption 1. The number of the observability constructed paths (3) of (2) is finite, while the length is $+\infty$.

Theorem 3.5. If a system (2) satisfies Assumption 1, then (2) is reconstructible iff each subsystem of the set $\{P_{t_0-\tau(t_0)}, P_{t_0-\tau(t_0)+1}, \ldots, P_{t_0}\}$ is reconstructible.

P r o o f. If a system (2) is reconstructible, then by Definition 3.1 and Assumption 1, each subsystem $P_{t_0-\tau(t_0)+i}$, $i = 0, \ldots, \tau(t_0)$ is of infinite length, and hence reconstructible.

Assume that for a system (2), each subsystem $P_{t_0-\tau(t_0)+i}$, $i = 0, \ldots, \tau(t_0)$, is reconstructible, then each observability constructed path of (2) is also reconstructible. For each observability constructed path i, there exists $T_i > 0$ s.t. $\forall t \geq T_i$, if t belongs to the time sequence of the path, then the state of (2) at time step t can be determined (by the corresponding input sequence and output sequence). Since there are finitely many observability constructed paths, we choose the maximum of all these T_i's, then after that time step, the states of (2) can be determined, i. e., (2) is reconstructible. □

Theorem 3.6. If a system (2) satisfies Assumption 1, then (2) is reconstructible if and only if system P_o is reconstructibility.

P r o o f. By Theorem 3.5, if system P_o is reconstructible, system $P_{t_0-\tau(t_0)+i}$ is reconstructible, $i = 0, 1, \ldots, \tau(t_0)$, then (2) is reconstructible. If system P_o is not reconstructible, then (2) is not reconstructible by Theorem 3.5 □

3.4. An algorithm to judge whether a BCN is reconstructable

Now we define the weighted pair graph for BCN (2).

Definition 3.7. (Zhang et al. [28]) Consider a BCN (2) with $\tau(t) \equiv 0$. A weighted digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}, 2^{\Delta_M})$, where $\mathcal{V}$ denotes the vertex set, $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ denotes the edge set, and $\mathcal{W} : \mathcal{E} \to 2^{\Delta_M}$ denotes the weight function, is called a weighted pair graph of BCN (2) if $\mathcal{V} = \{\{x, x'\} \in \Delta_N \times \Delta_N : x \neq x', Hx = Hx'\}$, for all $(x_1, x_1'), (x_2, x_2') \in \mathcal{V} \times \mathcal{V}$, $((x_1, x_1'), (x_2, x_2')) \in \mathcal{E}$ iff there exists $u \in \Delta_M$ such that $Lux_1 = x_2$ and $Lux_1' = x_2'$, or $Lux_1 = x_2'$ and $Lux_1' = x_2$, for all edges $e = ((x_1, x_1'), (x_2, x_2')) \in \mathcal{E}$, $\mathcal{W}(e) = \{u \in \Delta_M | Lux_1 = x_2$ and $Lux_1' = x_2'$, or, $Lux_1 = x_2'$ and $Lux_1' = x_2\}$.

According to Definition 3.1, to judge whether a BCN (2) is reconstructible, we need to check the set of all vertices of its weighted pair graph $(\mathcal{V}, \mathcal{E}, \mathcal{W})$.

Now we design an algorithm to construct a DFA for a System P_o according to its weighted pair graph. The new DFA is denoted by $A_{\mathcal{V}_n}$, and will be used to obtain the main results.

Based on [[28], Theorem 3.4] and Algorithm 1 the following theorem holds.

Theorem 3.8. If a system (2) satisfies Assumption 1, then its observability constructed path P_o is not reconstructable if and only if the DFA $A_{\mathcal{V}}$ generated by Algorithm 1 recognizes language $(\Delta_M)^*$, i. e., $L(A_{\mathcal{V}}) = (\Delta_M)^*$.

Algorithm 1 (Zhang et al. [28]) An algorithm for returning a DFA that determines the reconstructibility of the Observability constructed path P_o of BCN (2).

Input: The observability constructed path P_o of BCN (2) and its weighted pair graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$

Output: A DFA $A_{\mathcal{V}}$

$S := \{\mathcal{V}\}.S^1_{temp} := \mathcal{V}.S^2_{temp} := \emptyset$
while $S^1_{temp} \neq \emptyset$ **do**
  for all $s \in S^1_{temp}$ and $j \in [1, M]$ **do**
    $s_j := \{v_s \in \mathcal{V} |$ there is $v \in s$ such that $(v, v_s) \in \mathcal{E}$ and $\delta^j_M \in \mathcal{W}((v, v_s))\}$
    if $s_j \neq \emptyset$ and $s_j \notin S$ **then**
      $S := S \cup \{s_j\}.S^2_{temp} := S^2_{temp} \cup \{s_j\}.\tau(s, \delta^j_M) := s_j$
    else if $s_j \neq \emptyset$ **then**
      $\tau(s, \delta^j_M) := s_j$
    else
      τ is not well defined at (s, δ^j_M)
    end if
  end for
  $S^1_{temp} := S^2_{temp}.S^1_{temp} := \emptyset$
end while

P r o o f. Since for (2), the observability constructed path is of infinite length, the path coincides with (2) with $\tau(t) \equiv 0$, and the BCN considered in [[28], Theorem 3.4]. Hence this result holds. □

Based on Theorem 3.8 and Algorithm 1 the following theorem holds.

Theorem 3.9. If a system (2) satisfies Assumption 1, then A BCN (2) is not reconstructible if and only if for its observability constructed path P_o, the DFA $A_{\mathcal{V}}$ generated by Algorithm 1 recognizes language $(\Delta_M)^*$, i. e., $L(A_{\mathcal{V}}) = (\Delta_M)^*$.

P r o o f. Since the observability constructed path of the system (2) is of infinite length, an equivalent condition for the observability of the path is obtained by Theorem 3.8. In addition, by Theorem 3.6, the reconstructibility of the path is equivalent to the reconstructibility of (2). Hence this result holds. □

3.5. Determining the current state.

We have shown how to determine whether a given BCN (2) is reconstructible. One also sees that each U outside of $L(A_{\mathcal{V}})$ is a homing input sequence. Next we show how to use a homing input sequence to determine the current state.

Assume a reconstructible Boolean control network (2) satisfies Assumption 1. where the initial state sequence $(x(t_0 - \tau(t_0)), x(t_0 - \tau(t_0) + 1), \ldots, x(t_0)) \in \mathcal{D}^{n(\tau(t_0)+1)}$ is given and unknown. By Theorem 3.9. the language recognized by the DFA $A_{\mathcal{V}}$ generated by Algorithm 1 is a proper subset of $(\Delta_M)^*$. Then for each given homing input sequence $U = u_0 \ldots u_{p-1} \in (\Delta_M)^* \backslash L(A_{\mathcal{V}})$ and the corresponding output sequence $Y = y(t_0 -$

$\tau(t_0)), y(t_0 - \tau(t_0) + 1), \ldots, y(t_0), y(t_1), \ldots, y(t_T) \in (\Delta_Q)^{p+1}$, Algorithm 3.10 returns the current state $x(t_T)$. $\mathcal{X}_0$ in Algorithm 3.10 contains all possible states producing output y_0 and hence contains x_0. At each time step $1 \leq t \leq p$. $\mathcal{X}_t$ contains all states that are driven from initial state x_0 by input sequence $u_0 \ldots u_{t-1}$ and correspond to output sequence $y_0 \ldots y_t$. Then by Theorem 3.9. $\mathcal{X}_p$ is a signleton. and the unique element of $\mathcal{X}_p$ is the current state.

Algorithm 3.10. 1. Based on Definition 3.4. find the output sequence $Y = y_0 \ldots y_p$ of an observability path $P_{t_0-\tau(t_0)+i}$ of the tree $T_{t_0-\tau(t_0)+i}$ that includes vertice $x(t_0 + T)$ and the length p of a shortest observability path $P_{t_0-\tau(t_0)+i}$, $i = 0, 1, \ldots, \tau(t_0)$.

2. set $\mathcal{X}_0 := \{x | x \in \Delta_N, Hx = y_0\}$, $k = 0$.

3. set $\mathcal{X}_{k+1} = \{x | x \in \Delta_N, HLu_i x = y_{i+1}\}$, $k = k + 1$.

4. if $k = p$ stop. Else go back to Step 3.

4. ILLUSTRATIVE EXAMPLES

The following Example 4.1 illustrates the constructed forest, the weighted pair graph, how Algorithm 1 works, and how to use a homing input sequence to determine the current state

Example 4.1. Consider the following Boolean control network:

$$\begin{aligned} x(t+1) &= \delta_4[3,1,2,4,1,3,4,2]u(t)x(t-\tau(t)), \\ y(t) &= \delta_2[1,2,1,1]x(t), \end{aligned} \tag{5}$$

where $t = 0, 1, \ldots$, $x \in \Delta_4$, $y, u \in \Delta$,

$$\tau(t) = \begin{cases} \tau(t) = \dfrac{t+1}{2}, & t \text{ is odd and } t_0 \leq t \leq t_0 + 13 \\ \tau(t) = \dfrac{t}{2}, & t \text{ is even and } t_0 \leq t \leq t_0 + 13 \\ \tau(t) = 7, & t \geq t_0 + 14 \end{cases} \tag{6}$$

The constructed forest of (5) shown in Figure 2 satisfies Assumption 1. The weighted pair graph of the observability constructed path of the Boolean control network (5) is show in Figure 3. Putting the constructed forest and weighted pair graph into Algorithm 1, Algorithm 1 returns a DFA $A_{\mathcal{V}}$. From Figure 4 it follows that $A_{\mathcal{V}}$ is not complete, then by Theorem 3.9, the Boolean control network (5) is reconstructible.

Base on Algorithm 3.10, find the output sequences $Y = y(0)y(2)y(6)$ of an observability path of the tree in Figure 2 that includes vertice $x(6)$ and the length $p + 1 = 7$. Note that $\delta_2^2 * \delta_2^2 * * \notin L(A_{\mathcal{V}})$ where $*$ represent δ_2^0 or δ_2^1 by Figure 4; then $\delta_2^2 * \delta_2^2 * *$ is a homing input sequence. Next we use $\delta_2^2 * \delta_2^2 * *$ to determine the current state $x(6)$. Choose as an unknown initial state $x(0) = \delta_4^1$. Then the output sequence is $y(0) = 1, y(1) = 1, y(2) = 1, y(3) = 1, y(4) = 0, y(5) = 3, y(6) = 2$. According to Algorithm 3.10, choose the observability constructed path is $0 \to 2 \to 6 \to 14 \to 22 \ldots$ and the length is 3, then $y(0) = 1, y(2) = 1, y(6) = 2$, $\mathcal{X}_0 = \{\delta_4^1, \delta_4^3, \delta_4^4\}$, $\mathcal{X}_1 = \{\delta_4^1, \delta_4^4\}$, $\mathcal{X}_2 = \{\delta_4^4\}$. Hence the current state is $x(6) = \delta_4^4$.

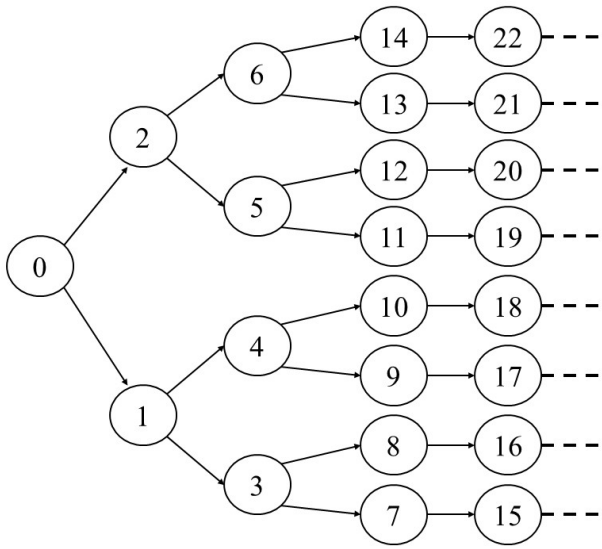

Fig. 2. Constructed forest of (5) with the time delay function (6) where the number in each circle denotes the time step.

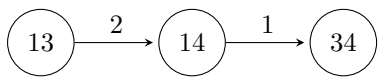

Fig. 3. Weighted pair graph of Boolean control network (5), where number ij in each circle denotes state pair $\{\delta_4^i, \delta_4^j\}$, and number k beside each edge denotes weight $\{\delta_2^k\}$ of the corresponding edge.

The following Example 4.2 shows that without Assumption 1, None of the Theorem 3.5, Theorem 3.6 and Theorem 3.9 holds.

Example 4.2. Consider the following Boolean control network:

$$\begin{aligned} x(t+1) &= \delta_4[3,2,4,1,1,2,4,2]u(t)x(t-\tau(t)) \\ y(t) &= \delta_2[1,2,2,1]x(t) \end{aligned} \tag{7}$$

where $t = 0, 1, \ldots$, $x \in \Delta_4$, $y, u \in \Delta$,

$$\begin{cases} \tau(t) = t - f(\mathfrak{m}(t+1), \mathfrak{n}(t+1) - 1), & \text{if } \mathfrak{n}(t+1) > t \\ \tau(t) = t, & \text{otherwise,} \end{cases} \tag{8}$$

where $f(\mathfrak{m}, \mathfrak{n}) = C_{\mathfrak{n}+1}^2 + (\mathfrak{m}-1)n + C_{\mathfrak{m}-1}^2$,

$$\begin{cases} \mathfrak{m}(t) = t - C_{\mathfrak{s}(t)}^2, \\ \mathfrak{n}(t) = \mathfrak{s}(t) - \mathfrak{m}(t), \\ \mathfrak{s}(t) = \lfloor \sqrt{2t - \frac{3}{4}} + \frac{3}{2} \rfloor \end{cases}$$

and $\lfloor \cdot \rfloor$ represents the maximal integer no greater than $\cdot$.

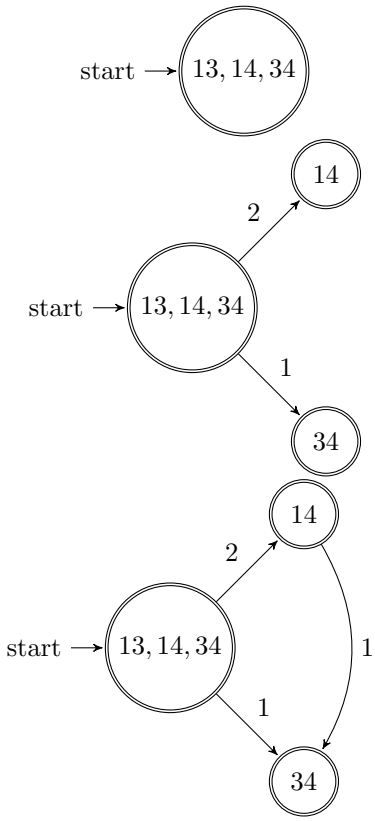

Fig. 4. The DFA $A_{\mathcal{V}}$ with respect to Boolean control networks (5) generated process by Algorithm 1, where the number ij in each circle denotes the state pair (δ_4^i, δ_4^j), and the weight k beside each edge denotes the input δ_2^k.

The delay function (8) contains infinitely many observability constructed paths and the length of each of the observability constructed paths is infinite (cf. Figure 5), which does not satisfies Assumption 1. Arbitrarily select one of the observability constructed paths. The subsystem is with no delay and its form is shown in (4). Assume that the subsystem is reconstructible, then there exists time T (e. g. $T = 3$), and the state of the subsystem at time T can be determined by an input sequence and the corresponding output sequence while the state at $T = 1$ cannot be determined, so the state at $t = 1, 2, 4, 7, \ldots$ (that is, the time in the first column in Figure 5) cannot be determined. Therefore, the system (7) can not be reconstructible.

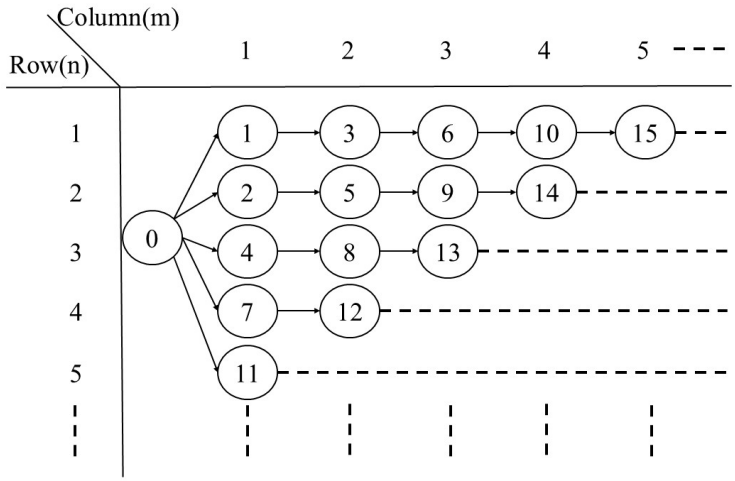

Fig. 5. Constructed forest of (7) with the time delay function (8) where the number in each circle denotes the time step.

5. CONCLUSION

In this paper, the reconstructibility of Boolean control networks with time delay in states was studied under a mild assumption based on the theories of finite automata and formal languages, weight pair graph and constructed forest. Based on this approach, an equivalent test criterion has been given, and then an algorithm has been designed to determine the reconstructibility and another algorithm has been designed to determine the current state.

ACKNOWLEDGEMENT

This work was supported by National Natural Science Foundation (NNSF) of China under Grant Nos. 61573288 and 61603109.

REFERENCES

[1] T. Akutsu, M. Hayashida, W. Ching, and M. K. Ng: Control of Boolean networks: Hardness results and algorithms for tree structured networks. J. Theoret. Biology *244* (2007), 670–679. DOI:10.1016/j.jtbi.2006.09.023

[2] A. Bolotin: Constructibility of the universal wave function. Foundat. Physics *46* (2016), 1–16. DOI:10.1007/s10701-016-0018-7

[3] D. Cheng: On semi-tensor product of matrices and its applications. Acta Math. Appl. Sinica *19* (2003), 219–228. DOI:10.1007/s10255-003-0097-z

[4] D. Cheng: Input-state approach to Boolean networks. IEEE Trans. Neural Networks. *20* (2009), 512–521. DOI:10.1109/tnn.2008.2011359

[5] D. Cheng and H. Qi: A linear representation of dynamics of Boolean networks. IEEE Trans. Automat. Control *55* (2010), 2251–2258. DOI:10.1109/tac.2010.2043294

[6] D. Cheng, H. Qi, and Z. Li: Realization of Boolean control networks. Automatica *46* (2010), 62–69. DOI:10.1016/j.automatica.2009.10.036

[7] D. Cheng, H. Qi, and Z. Li: Controllability and Observability of Boolean Control Networks. Springer-Verlag, London 2011.

[8] D. Cheng, H. Qi, and Z. Li: Analysis and Control of Boolean Networks. Springer–Verlag, London 2011.

[9] C. Farrow et al.: Scalar equations for synchronous Boolean networks with biological applications. IEEE Trans. Neural Networks *15* (2004), 348–354. DOI:10.1109/tnn.2004.824262

[10] E. Fornasini and M. E. Valcher: Observability, reconstructibility and state observers of Boolean control networks. IEEE Trans. Automat. Control *58* (2013), 1390–1401. DOI:10.1109/tac.2012.2231592

[11] E. Fornasini and M. E. Valcher: Fault detection analysis of Boolean control networks. IEEE Trans. Automat. Control *60* (2015), 2734–2739. DOI:10.1109/tac.2015.2396646

[12] L. I. Haitao, G. Zhao, M. Meng, and J. Feng: A survey on applications of semi-tensor product method in engineering. Science China (Inform. Sci.) *61* (2018), 1, 010202. DOI:10.1007/s11432-017-9238-1

[13] M. Han, Y. Liu, and Y. Tu: Controllability of Boolean control networks with time delays both in states and inputs. Neurocomputing *129* (2014), 467–475. DOI:10.1016/j.neucom.2013.09.012

[14] T. Ideker, T. Galitski, and L. Hood: A new approach to decoding life: systems biology. Ann. Rev. Genomics Hum. Genet. *2* (2001), 343–372. DOI:10.1146/annurev.genom.2.1.343

[15] J. Kari: A Lecture Note on Automata and Formal Languages. http://users.utu.fi/jkari/automata/, 2016.

[16] S. A. Kauffman: Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theoret. Biology *22* (1969), 437–467. DOI:10.1016/0022-5193(69)90015-0

[17] H. K. Khalil: Nonlinear Systems. MacMillan, New York 1992.

[18] F. Li, J. Sun, and Q. Wu: Observability of Boolean control networks with state time delays. IEEE Trans. Neural Networks *22* (2011), 948–954. DOI:10.1109/tnn.2011.2126594

[19] F. Li, J. Sun, and Q. Wu: Observability of Boolean control networks with state time delays. IEEE Trans. Neural Networks *22* (2011), 948–954. DOI:10.1109/tnn.2011.2126594

[20] J. Lu, H. Li, Y. Liu, and F. Li: Survey on semi-tensor product method with its applications in logical networks and other finite-valued systems. IET Control Theory Appl. *11* (2017) 13, 2040–2047. DOI:10.1049/iet-cta.2016.1659

[21] H. Sui: Regulation of Cellular states in mammalian cells from a genomewide view. In: Gene Regulations and Metabolism – Postgenomic Computational Approaches (J. Collado-Vides, ed.), MIT Press, MA 2002, pp. 181–220.

[22] A. Thomasian: Reconstruct versus read-modify writes in RAID. Inform. Process. Lett. *93* (2005), 163–168. DOI:10.1016/j.ipl.2004.10.009

[23] A. Valmari: On constructibility and unconstructibility of LTS operators from other LTS operators. Acta Inform. *52* (2015), 207–234. DOI:10.1007/s00236-015-0217-2

[24] K. Zhang and L. Zhang: Observability of Boolean control networks: a unified approach based on the theories of finite automata. IEEE Trans. Automat. Control *61* (2015), 6854–6861.

[25] L. Zhang and K. Zhang: Controllability and observability of Boolean control networks with time-variant delays in states. IEEE Trans. Neural Networks Learning Syst. *24* (2013), 1478–1484. DOI:10.1109/tnnls.2013.2246187

[26] L. Zhang and K. Zhang: Controllability of time-variant Boolean control networks and its application to Boolean control networks with finite memories. Science China (Inform. Sci.) *56* (2013), 1–12. DOI:10.1007/s11432-012-4651-2

[27] K. Zhang and L. Zhang: Controllability of probabilistic Boolean control networks with time-variant delays in states. Science China (Inform. Sci.) *59* (2016), 092204:1–092204:10. DOI:10.1007/s11432-012-4651-2

[28] K. Zhang, L. Zhang, and R. Su: A weighted pair graph representation for reconstructibility of Boolean control networks. SIAM J. Control Optim. *54* (2016), 3040–3060. DOI:10.1137/140991285

[29] Y. Zhao, H. Qi, and D. Cheng: Input-state incidence matrix of Boolean control networks and its applications. Systems Control Lett. *59* (2010), 767–774. DOI:10.1016/j.sysconle.2010.09.002

Ping Sun, College of Automation, Harbin Engineering University, Harbin, 150001. P. R. China.
e-mail: hrbeusp@126.com

Lijun Zhang, College of Automation, Harbin Engineering University, Harbin, 150001. P. R. China.
School of Marine Technology, Northwestern Polytechnical University, Xi'an, 710072. P. R. China.
e-mail: zhanglj7385@nwpu.edu.cn

Kuize Zhang, College of Automation, Harbin Engineering University, Harbin, 150001. P. R. China.
e-mail: zkz0017@163.com