Proceedings of the 2nd International Workshop on Biological Processes & Petri Nets (BioPPN2011)
online: http://ceur-ws.org/Vol-724 pp.7-20

Optimal Control of Asynchronous Boolean Networks Modeled by Petri Nets

Koichi Kobayashi and Kunihiko Hiraishi

Japan Advanced Institute of Science and Technology, Ishikawa 923-1292, Japan
Tel:+81-761-51-1282, Fax:+81-761-51-1149
{k-kobaya,hira}@jaist.ac.jp

Abstract. A Boolean network model is one of the models of gene regulatory networks, and is widely used in analysis and control. Although a Boolean network is a class of discrete-time nonlinear systems and expresses the synchronous behavior, it is important to consider the asynchronous behavior. In this paper, using a Petri net, a new modeling method of asynchronous Boolean networks with control inputs is proposed. Furthermore, the optimal control problem of Petri nets expressing asynchronous Boolean networks is formulated, and a solution method is proposed. The proposed approach provides us a new control method of gene regulatory networks.

1 Introduction

In recent years, there have been a lot of studies on modeling, analysis, and control of gene regulatory networks in both the control community and the theoretical biology community. Gene regulatory networks are in general expressed by ordinary/partial differential equations with high nonlinearity and high dimensionality. In order to deal with such a system, it is important to consider a simple model, and various models such as Bayesian networks, Boolean networks, hybrid systems (piecewise affine models), and Petri nets have been developed so far (see e.g., [14]). In control problems, Boolean networks and hybrid systems are frequently used [1, 3, 4, 17, 18]. However, in the hybrid systems-based approach, a class of gene regulatory networks are limited to low-dimensional systems, because the computation time to solve the control problem is too long. In Boolean networks, dynamics such as interactions between genes are expressed by Boolean functions [15]. Although there is a criticism that a Boolean network is too simple as a model of gene regulatory networks, this model can be relatively applied to large-scale systems. Furthermore, since the behavior of gene regulatory networks is probabilistic by the effects of noise, a probabilistic Boolean network (PBN) has been proposed in [20].

Although a Boolean network is a class of discrete-time nonlinear systems and expresses the synchronous behavior, it is important to consider the asynchronous behavior. Asynchronous Boolean networks have been proposed in [12, 22]. In [12], we assume that the updating time of the concentration level of each gene is given in advance. In [22], asynchronous Boolean networks are modeled by non-deterministic dynamical systems. Furthermore, the asynchronous behavior is expressed as the probabilistic behavior. However, in these two methods, the asynchronous behavior is not directly modeled.

On the other hand, a Petri net is well known as a model expressing the asynchronous behavior [25]. A Petri net is a class of directed bipartite graphs, in which the nodes represent transitions and places. The methods to express asynchronous Boolean networks as Petri nets have been proposed in [8, 21]. However, in these methods, the control input is not considered, and these methods cannot be directly applied to the control problem. The control input in biological networks has the following significance. For example, the value of the control input expresses whether a stimulus is given to a cell. Then the control input is designed to obtain the state trajectory that transits from the initial state to the desired one. So the control input can represent the current status of therapeutic interventions, which are realized by radiation, chemotherapy, and so on. In order to develop gene therapy technologies (see e.g., [16, 19]) in future, it is important to consider control methods of Boolean networks.

Thus in this paper, the optimal control problem of asynchronous Boolean networks modeled by Petri nets is discussed. First, based on the method proposed in [8] and the notation of external input places [13, 23], we propose a new method to transform asynchronous Boolean networks with control inputs into Petri nets with external input places. Furthermore, the obtained Petri net is transformed into a logical dynamical system, which is a class of linear systems with binary states and binary inputs. Next, the optimal control problem is formulated, and the biological significance is also discussed by using a simple example. Finally, the solution method of the optimal control problem is proposed. In the proposed solution method, the optimal control problem is reduced to an integer linear programming (ILP) problem. The proposed approach provides us a new control method of gene regulatory networks.

This paper is organized as follows. In Section 2, synchronous Boolean networks and asynchronous Boolean networks are introduced. In Section 3, Petri nets expressing asynchronous Boolean networks are derived. In Section 4, we explain the method to transform the obtained Petri net into a logical dynamical system. In Section 5, the optimal control problem is formulated, and in Section 6, a solution method is proposed. In Section 7, we conclude this paper.

Notation: Let $\mathcal{R}$ denote the set of real numbers. Let $\{0,1\}^{m\times n}$ denote the set of $m \times n$ matrices, which consists of elements 0 and 1. For the finite set M, let $|M|$ denote the number of elements. Let I_n and $0_{m\times n}$ denote the $n \times n$ identity matrix and the $m \times n$ zero matrix, respectively. For simplicity of notation, we sometimes use the symbol 0 instead of $0_{m\times n}$, and the symbol I instead of I_n. For a matrix M, let M^T denote the transpose of M.

2 Synchronous/Asynchronous Boolean Networks

First, we explain synchronous Boolean networks (SBNs).

Consider the following SBN:

$$x(k+1) = f_a(x(k)) \tag{1}$$

where $x \in \{0,1\}^n$ is the state (e.g., the concentration of genes), $k = 0, 1, 2, \ldots$ is the discrete time. $f_a : \{0,1\}^n \to \{0,1\}^n$ is a given Boolean function with logical operators such as AND ($\wedge$), OR ($\vee$), and NOT ($\neg$). Since the SBN (1) is deterministic, $x(k+1)$ is uniquely determined for a given $x(k)$.

To consider the control problems, we add the control input to the SBN (1) as follows:

$$x(k+1) = f(x(k), u(k)) \tag{2}$$

where $u \in \{0,1\}^m$ is the control input, i.e., the value of u (e.g., the concentration of genes) can be arbitrarily given, and $f : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ is a given Boolean function. The i-th element of the state x, the i-th element of the control input u and the i-th element of the Boolean function f are denoted by x_i, u_i and f_i, respectively. Also in the SBN (2), $x(k+1)$ is uniquely determined for given $x(k)$ and $u(k)$.

Next, we explain asynchronous Boolean networks (ABNs). Some methods for expressing ABNs have proposed so far [12, 22]. In this section, we explain a method in [22] for expressing ABNs as nondeterministic systems. A Petri net-based approach will be explained in Section 3. In the case that ABNs are expressed as nondeterministic systems, the behavior of ABNs is obtained by the union of the behaviors of the following n SBNs:

$$\Sigma_i : \begin{cases} x_i(k+1) = f_i(x(k), u(k)), \\ x_j(k+1) = x_j(k), \quad \forall j \in \{1, 2, \ldots, n\} \setminus \{i\}, \end{cases} \tag{3}$$

where $i = 1, 2, \ldots, n$.

We show an example of SBNs and ABNs.

Example 1. As a simple example, consider the following SBN of an apoptosis network [9]:

$$\begin{cases} x_1(k+1) = \neg x_2(k) \wedge u(k), \\ x_2(k+1) = \neg x_1(k) \wedge x_3(k), \\ x_3(k+1) = x_2(k) \vee u(k) \end{cases} \tag{4}$$

where the concentration level (high or low) of the inhibitor of apoptosis proteins (IAP) is denoted by x_1, the concentration level of the active caspase 3 (C3a) by x_2, and the concentration level of the active caspase 8 (C8a) by x_3. The concentration level of the tumor necrosis factor (TNF, a stimulus) is denoted by u, and is regarded as the control input.

In the case of synchronous Boolean dynamics, state transitions can be computed by directly using (4). For example, for $x(0) = [\ 1 \ \ 1 \ \ 1\ ]^T$ and $u(k) = 0$, we obtain $x(1) = [\ 0 \ \ 0 \ \ 1\ ]^T$. By computing the transition from each state, we obtain the state transition diagram in Fig. 1 (left). In Fig. 1 (left), the number assigned to each node denotes x_1, x_2, x_3 (elements of the state),

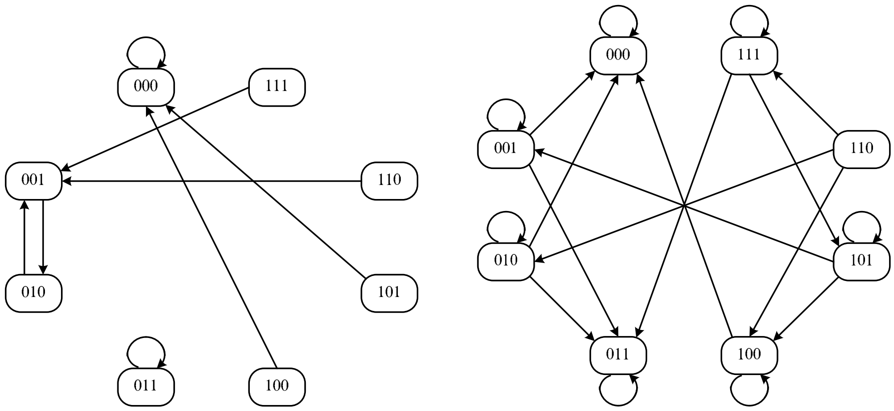

Fig. 1. (Left) State transition diagram of (4) and $u(k) = 0$, (Right) State transition diagram of (5), (6), (7) and $u(k) = 0$.

In the case of asynchronous Boolean dynamics, we consider the following three SBNs

$$\Sigma_1 : \begin{cases} x_1(k+1) = \neg x_2(k) \wedge u(k), \\ x_2(k+1) = x_2(k), \\ x_3(k+1) = x_3(k), \end{cases} \tag{5}$$

$$\Sigma_2 : \begin{cases} x_1(k+1) = x_1(k), \\ x_2(k+1) = \neg x_1(k) \wedge x_3(k), \\ x_3(k+1) = x_3(k) \end{cases} \tag{6}$$

$$\Sigma_3 : \begin{cases} x_1(k+1) = x_1(k), \\ x_2(k+1) = x_2(k), \\ x_3(k+1) = x_2(k) \vee u(k) \end{cases} \tag{7}$$

State transitions can be computed by using (5), (6), (7). For example, for $x(0) = [\ 1\ \ 1\ \ 1\ ]^T$ and $u(k) = 0$, we obtain $x(1) = \{[\ 0\ \ 1\ \ 1\ ]^T, [\ 1\ \ 0\ \ 1\ ]^T, [\ 1\ \ 1\ \ 1\ ]^T\}$. In a similar way, by computing the transition from each state, we obtain the state transition diagram in Fig. 1 (right).

Comparing the left figure with the right figure in Fig. 1, we see that a part of behaviors is clearly different. □

In this paper, ABNs are modeled by Petri nets, not multiple SBNs. By using Petri nets, we can consider several situations. For example, although each x_i is independently activated in (3), activation of combinations of x_i can be considered.

3 Transformation of Boolean Networks into Petri Nets

Using each Boolean function f_i in the SBN (2), consider to express an ABN as a Petri net. Based on a complementary-place transformation, a Petri net expressing an ABN has been proposed in [8], but the control input has not been considered. In this paper, as an extension of the method in [8], we propose a modeling method of a Petri net expressing an ABN with the control input.

First, some notations are prepared. By $\mathcal{I}(j)$, $j = 1, 2, \ldots, n$, denote the state and the control input included in the Boolean function f_j. In the example of (4), we obtain $\mathcal{I}(1) = \{x_2, u\}$, $\mathcal{I}(2) = \{x_1, x_3\}$, and $\mathcal{I}(3) = \{x_2, u\}$. Next, we define a logical parameter $K_j(X) \in \{0, 1\}$, $X \subseteq \mathcal{I}(j)$, $j = 1, 2, \ldots, n$. If the value of each element included in X is '1', and the value of the other element is '0', then either $K_j(X) = 1$ or $K_j(X) = 0$ is determined. In the example of (4), we obtain

$$K_1(\emptyset) = 0, \quad K_1(\{x_2\}) = 0, \quad K_1(\{u\}) = 1, \quad K_1(\{x_2, u\}) = 0,$$
$$K_2(\emptyset) = 0, \quad K_2(\{x_1\}) = 0, \quad K_2(\{x_3\}) = 1, \quad K_2(\{x_1, x_3\}) = 0,$$
$$K_3(\emptyset) = 0, \quad K_3(\{x_2\}) = 1, \quad K_3(\{u\}) = 1, \quad K_3(\{x_2, u\}) = 1.$$

Next, consider to derive a Petri net expressing Boolean networks. In the derived Petri net, the number of places is given as $2(n + m)$, that is, for each x_i in (2), two places x_i and $\overline{x}_i$ are prepared. In a similar way, for each u_i in (2), two places u_i and $\overline{u}_i$ are prepared. $\overline{x}_i$ and $\overline{u}_i$ are called complementary places [8]. The number of transitions is given as $\sum_{i=1}^{n} 2^{|\mathcal{I}(i)|}$. In the example of (4), the number of transitions is given as $2^2 + 2^2 + 2^2 = 12$. From the property of Boolean networks, the following assumptions are made.

Assumption 1 *The maximum number of tokens in each place is equal to* 1*.*

Assumption 2 *A sum of the number of tokens in x_i (u_i) and that in $\overline{x}_i$ ($\overline{u}_i$) is equal to* 1*.*

In addition, suppose that u_i and $\overline{u}_i$ are given as an external input place [13, 23]. In u_i and $\overline{u}_i$, a token is arbitrary generated, but the above two assumptions must be satisfied.

Under the above preparations, we define a Petri net expressing an ABN. In [8], the Petri net expressing an ABN without the control input is defined. The following definition gives the Petri net expressing an ABN with the control input, and is an extension of the definition in [8].

Definition 1. *For a given SBN (2), the Petri net expressing an ABN is defined as follows:*

$$N_c = (P \cup P_c, T, Pre, Post) \tag{8}$$

where

- $P = \{x_1, \overline{x}_1, x_2, \overline{x}_2, \ldots, x_n, \overline{x}_n\}$ *is the set of places,*
- $P_c = \{u_1, \overline{u}_1, u_2, \overline{u}_2, \ldots, u_m, \overline{u}_m\}$ *is the set of external input places,*
- $T = \{t_{x_i, X}, i = 1, 2, \ldots, n, X \subseteq \mathcal{I}(i)\}$ *is the set of transitions,*
- $Pre : (P \cup P_c) \times T \to \{0, 1\}$ *is the mapping defining arcs between places and transitions,*
- $Post : T \times P \to \{0, 1\}$ *is the mapping defining arcs between transitions and transitions.*

The functions Pre and Post are defined as follows:
(i) Case of $x_i \notin \mathcal{I}(i)$ *(*x_i *is not a self-regulator): For a given transition* $t_{x_i,X}$*, the following terms are defined (all the other terms are equal to zero):*

$$\begin{aligned}
&Pre(x_i, t_{x_i,X}) = Post(t_{x_i,X}, \overline{x}_i) = 1 - K_i(X),\\
&Pre(\overline{x}_i, t_{x_i,X}) = Post(t_{x_i,X}, x_i) = 1 - K_i(X),\\
&Pre(x_j, t_{x_i,X}) = Post(t_{x_i,X}, x_j) = 1, \quad \forall x_j \in X,\\
&Pre(\overline{x}_j, t_{x_i,X}) = Post(t_{x_i,X}, \overline{x}_j) = 1, \quad \forall x_j \in \mathcal{I}(i) - X,\\
&Pre(u_j, t_{x_i,X}) = 1, \quad \forall u_j \in X,\\
&Pre(\overline{u}_j, t_{x_i,X}) = 1, \quad \forall u_j \in \mathcal{I}(i) - X.
\end{aligned}$$

(ii) Case of $x_i \in \mathcal{I}(i)$ *(*x_i *is a self-regulator): Consider a given transition* $t_{x_i,X}$*. if* $x_i \in X$*, then only the case of* $K_i(X) = 0$ *is considered. Therefore, the following terms are defined:*

$$\begin{aligned}
&Pre(x_i, t_{x_i,X}) = Post(t_{x_i,X}, \overline{x}_j) = 1,\\
&Pre(x_j, t_{x_i,X}) = Post(t_{x_i,X}, x_j) = 1, \quad \forall x_j \in X, \quad x_j \neq x_i,\\
&Pre(\overline{x}_j, t_{x_i,X}) = Post(t_{x_i,X}, \overline{x}_j) = 1, \quad \forall x_j \in \mathcal{I}(i) - X,\\
&Pre(u_j, t_{x_i,X}) = 1, \quad \forall u_j \in X,\\
&Pre(\overline{u}_j, t_{x_i,X}) = 1, \quad \forall u_j \in \mathcal{I}(i) - X.
\end{aligned}$$

if $x_i \notin X$*, then only the case of* $K_i(X) = 1$ *is considered. Therefore, the following terms are defined:*

$$\begin{aligned}
&Pre(\overline{x}_i, t_{x_i,X}) = Post(t_{x_i,X}, x_j) = 1,\\
&Pre(x_j, t_{x_i,X}) = Post(t_{x_i,X}, x_j) = 1, \quad \forall x_j \in X,\\
&Pre(\overline{x}_j, t_{x_i,X}) = Post(t_{x_i,X}, \overline{x}_j) = 1, \quad \forall x_j \in \mathcal{I}(i) - X, \quad x_j \neq x_i,\\
&Pre(u_j, t_{x_i,X}) = 1, \quad \forall u_j \in \mathcal{I}(i) - X,\\
&Pre(\overline{u}_j, t_{x_i,X}) = 1, \quad \forall u_j \in X.
\end{aligned}$$

In the above definition, a sum of the number of tokens in u_i and that in $\bar{u}_i$ becomes zero by firing some transition. In this case, to satisfy Assumption 1 and Assumption 2, a token is generated in either u_i or $\bar{u}_i$.

We show a simple example.

Example 2. Consider the following simple SBN:

$$\begin{cases} x_1(k+1) = x_2(k), \\ x_2(k+1) = u(k) \end{cases} \tag{9}$$

From $x_1(k+1) = x_2(k)$, we obtain $K_1(\emptyset) = 0$ and $K_1(\{x_2\}) = 1$. In a similar way, from $x_1(k+1) = u(k)$, we obtain $K_2(\emptyset) = 0$ and $K_1(\{u\}) = 1$. Then we consider four transitions $t_{x_1,\emptyset}$, $t_{x_1,\{x_2\}}$, $t_{x_2,\emptyset}$, and $t_{x_2,\{u\}}$. We denote these

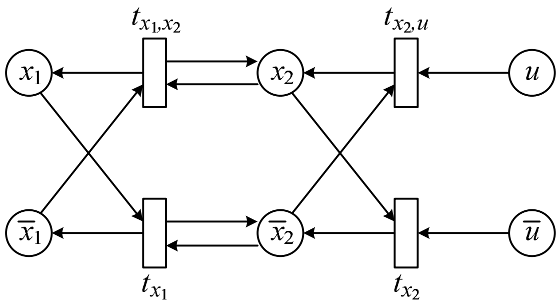

Fig. 2. Petri net expressing an ABN

transitions by t_{x_1}, t_{x_1,x_2}, t_{x_2}, and $t_{x_2,u}$, respectively. Then we obtain the Petri net in Fig. 2. In addition, *Pre* and *Post* in (8) are obtained as follows:

$$Pre = \begin{bmatrix} & t_{x_1} & t_{x_1,x_2} & t_{x_2} & t_{x_2,u} \\ x_1 & 1 & 0 & 0 & 0 \\ \bar{x}_1 & 0 & 1 & 0 & 0 \\ x_2 & 0 & 1 & 1 & 0 \\ \bar{x}_2 & 1 & 0 & 0 & 1 \\ u & 0 & 0 & 0 & 1 \\ \bar{u} & 0 & 0 & 1 & 0 \end{bmatrix}, \quad Post = \begin{bmatrix} & x_1 & \bar{x}_1 & x_2 & \bar{x}_2 \\ t_{x_1} & 0 & 1 & 0 & 1 \\ t_{x_1,x_2} & 1 & 0 & 1 & 0 \\ t_{x_2} & 0 & 0 & 0 & 1 \\ t_{x_2,u} & 0 & 0 & 1 & 0 \end{bmatrix}.$$

Suppose that one token is included in place $\bar{x}_1$, $\bar{x}_2$, and u. Then the transition $t_{x_2,u}$ may fire. If the transition $t_{x_2,u}$ fire, then one token is moved from $\bar{x}_2$ and u to x_2. A pair of x_2 and $\bar{x}_2$ satisfies Assumption 1 and Assumption 2, but a pair of u and $\bar{u}$ does not satisfy Assumption 2. So one token must be added in either u and $\bar{u}$ with fire. □

4 Transformation of Petri Nets into Logical Dynamical Systems

To consider the optimal control problem, it is desirable to transform a Petri net (8) into some linear form. In this section, based on a framework on modeling of hybrid dynamical systems [5], logical dynamical systems expressing Petri nets are derived.

Logical dynamical systems are given as a pair of linear state equations and linear inequality constraints with binary state variables and binary control input variables. A general form of logical dynamical systems is defined as follows.

Definition 2. *A logical dynamical system is given as*

$$x(k+1) = Ax(k) + Bv(k), \quad (10)$$

$$Cx(k) + Dv(k) \leq E \quad (11)$$

where $x(k) \in \{0,1\}^{n_d}$ *is the state variable, and* $v(k) \in \{0,1\}^{m_d}$ *is the input variable including auxiliary variables,* In modeling of hybrid dynamical systems, a mixed logical dynamical (MLD) system [5] is well known. The MLD system can be derived by replacing $x(k) \in \{0,1\}^{n_d}$ and $v(k) \in \{0,1\}^{m_d}$ in (10), (11) with $x(k) \in \mathcal{R}^{n_c} \times \{0,1\}^{n_d}$ and $v(k) \in \mathcal{R}^{m_c} \times \{0,1\}^{m_d}$. Since a Petri net is a class of discrete event systems, the MLD system is not used, and the logical dynamical system (10), (11) is used.

Let us consider to transform the Petri net (8) into the logical dynamical system (10), (11). Some notations are prepared. By $x_i(k), \bar{x}_i(k), u_i(k), \bar{u}_i(k) \in \{0,1\}$, denote existence or non-existence of a token in place $x_i, \bar{x}_i, u_i, \bar{u}_i \in \{0,1\}$ at time k. k may be regarded as the k-th firing in a firing sequence. Since $x_i(k), \bar{x}_i(k), u_i(k), \bar{u}_i(k)$ are binary variables, Assumption 1 satisfies. To satisfy Assumption 2, $x_i(k) + \bar{x}_i(k) = 1$ and $u_i(k) + \bar{u}_i(k) = 1$ are imposed. Next, by $t_{x_i,X_j}(k) \in \{0,1\}$, $i = 1,2,\ldots,n$, $j = 1,2,\ldots,2^{|\mathcal{I}(i)|}$, denote fire in the transition t_{x_i,X_j}. If $t_{x_i,X_j}(k) = 1$, then the transition t_{x_i,X_j} fires at time k. Otherwise, t_{x_i,X_j} does not fire. By using these notations, Petri nets (8) is transformed into logical dynamical systems.

First, a simple example is shown.

Example 3. Consider the Petri net in Fig. 2. From the property of fire, we obtain the following system expressing the Petri net in Fig. 2:

$$\begin{aligned}
x_1(k+1) &= t_{x_1,x_2}(k)\bar{x}_1(k)x_2(k) + (1 - t_{x_1,x_2}(k))x_1(k) \\
&\quad - t_{x_1}(k)x_1(k)\bar{x}_2(k), && (12)\\
\bar{x}_1(k+1) &= t_{x_1}(k)x_1(k)\bar{x}_2(k) + (1 - t_{x_1}(k))\bar{x}_1(k) \\
&\quad - t_{x_1,x_2}(k)\bar{x}_1(k)x_2(k), && (13)\\
x_2(k+1) &= t_{x_1,x_2}(k)\bar{x}_1(k)x_2(k) + t_{x_2,u}(k)\bar{x}_2(k)u(k) \\
&\quad + (1 - t_{x_1,x_2}(k) - t_{x_2,u}(k))x_2(k) - t_{x_2}(k)x_2(k)\bar{u}(k), && (14)\\
\bar{x}_2(k+1) &= t_{x_1}(k)x_1(k)\bar{x}_2(k) + t_{x_2}(k)x_2(k)\bar{u}(k) \\
&\quad + (1 - t_{x_1}(k) - t_{x_2}(k))\bar{x}_2(k) \\
&\quad - t_{x_1}(k)x_1(k)\bar{x}_2(k) - t_{x_2,u}(k)\bar{x}_2(k)u(k). && (15)
\end{aligned}$$

If the number of firing transitions at each time is limited to 1, then the inequality condition

$$t_{x_1}(k) + t_{x_1,x_2}(k) + t_{x_2}(k) + t_{x_2,u}(k) \leq 1 \tag{16}$$

is imposed. Then $x_1(k), \bar{x}_1(k), x_2(k), \bar{x}_2(k) \in \{0,1\}$, $x_1(k) + \bar{x}_1(k) = 1$, and $x_2(k) + \bar{x}_2(k) = 1$ hold thanks to the condition (16), $u(k), \bar{u}(k) \in \{0,1\}$, $u(k) + \bar{u}(k) = 1$, and the initial condition $x_1(0), \bar{x}_1(0), x_2(0), \bar{x}_2(0) \in \{0,1\}$, $x_1(0) + \bar{x}_1(0) = 1$, $x_2(0) + \bar{x}_2(0) = 1$. The system (12)–(15) is a nonlinear system, and can be linearized by using Lemma 1 in Appendix A. For example, $z_1(k) = t_{x_1,x_2}(k)\bar{x}_1(k)x_2(k)$ is equivalent to

$$\begin{aligned}
&t_{x_1,x_2}(k) + \bar{x}_1(k) + x_2(k) - z_1(k) \leq 2, \\
&-t_{x_1,x_2}(k) - \bar{x}_1(k) - x_2(k) + 3z_1(k) \leq 0.
\end{aligned}$$

By applying Lemma 1 to the other terms, we can obtain the logical dynamical system expressing the Petri net in Fig. 2. □

From this example, we see that the Petri net (8) can be expressed by the logical dynamical system (10), (11). Furthermore, when the Petri net (8) is expressed by the logical dynamical system (10), (11), variables x, v are given as

$$
\begin{aligned}
x(k) &= [\ x_1(k)\ \ \bar{x}_1(k)\ \ x_2(k)\ \ \bar{x}_2(k)\ \ \cdots\ \ x_n(k)\ \ \bar{x}_n(k)\ ]^T, \\
v(k) &= [\ U(k)\ \ T(k)\ \ Z(k)\ ]^T, \\
U(k) &= [\ u_1(k)\ \ \bar{u}_1(k)\ \ u_2(k)\ \ \bar{u}_2(k)\ \ \cdots\ \ u_m(k)\ \ \bar{u}_m(k)\ ]^T, \\
T(k) &= \left[\ t_{x_1,X_1}(k)\ \ \cdots\ \ t_{x_1,X_{2^{|\mathcal{I}(1)|}}}(k)\ \ \cdots\ \ t_{x_n,X_1}(k)\ \ \cdots\ \ t_{x_n,X_{2^{|\mathcal{I}(n)|}}}(k)\ \right]^T
\end{aligned}
$$

where $Z(k)$ is a auxiliary binary variable obtained by applying Lemma 1. Matrices/vectors A, B, C, D, E in (10), (11) can be derived from $Pre, Post$ in the Petri net (8) and Lemma 1.

Remark 1. One of the simple methods for modeling of ABNs is to express ABNs as switched systems with 2^n subsystems. 2^n subsystems are derived by all combinations of Boolean functions $f_1, f_2, \ldots, f_n$ in (2). In the case of (9) in Example 2, the following $2^n = 4$ subsystems:

$$
\begin{aligned}
\Sigma_1 &: x_1(k+1) = x_1(k), \quad x_2(k+1) = x_2(k), \\
\Sigma_2 &: x_1(k+1) = x_2(k), \quad x_2(k+1) = x_2(k), \\
\Sigma_3 &: x_1(k+1) = x_1(k), \quad x_2(k+1) = u(k), \\
\Sigma_4 &: x_1(k+1) = x_2(k), \quad x_2(k+1) = u(k)
\end{aligned}
$$

are obtained. From these subsystems, we can obtain the following system expressing an ABN:

$$x_1(k+1) = (\delta_1(k) + \delta_3(k))x_1(k) + (1 - \delta_1(k) - \delta_3(k))x_2(k), \tag{17}$$

$$x_2(k+1) = (\delta_1(k) + \delta_2(k))x_2(k) + (1 - \delta_1(k) - \delta_2(k))u(k) \tag{18}$$

where $\delta_1(k), \delta_2(k), \delta_3(k)$ are binary variables satisfying $\delta_1(k)+\delta_2(k)+\delta_3(k) \leq 1$, and correspond to $\Sigma_1, \Sigma_2, \Sigma_3$, respectively. Furthermore, $z_1 := \delta_1 x_1$, $z_2 := \delta_3 x_1$, $z_3 := \delta_1 x_2$, $z_4 := \delta_3 x_2$, $z_5 := \delta_2 x_2$, $z_6 := \delta_1 u$, and $z_7 := \delta_2 u$ are defined, and Lemma 1 is applied to $z_1, z_2, \ldots, z_7$. Thus we can obtain the logical dynamical system (10), (11). This method is called here a direct approach.

Comparing (17), (18) with (12)–(15), we see that the system (17), (18) is simpler than the system (12)–(15). However, for general cases, this fact does not hold. Here, we focus on the dimension of binary variables to switch Boolean functions. In (12)–(15), this dimension corresponds to the dimension of binary variables assigned to transitions, and is given as 4. In (17), (18), this dimension is given as 3. In general, in the direct approach, the dimension of binary variables to switch Boolean functions is given as $2^n - 1$. In the proposed Petri net-based approach, this dimension, i.e., $|T(k)|$ is given as $\sum_{i=1}^{n} 2^{|\mathcal{I}(i)|}$. In real gene regulatory networks, it is well known that $|\mathcal{I}(i)|$ is relatively smaller than n (see e.g., [2]). For example, in the case of $n = 10$ and $|\mathcal{I}(i)| = 3$, $2^n - 1 = 1023$ and $\sum_{i=1}^{n} 2^{|\mathcal{I}(i)|} = 80$ are obtained. Thus, in modeling of real gene regulatory networks, it is not appropriate to use the direct approach, and the proposed Petri net-based method provides us a simpler modeling method of ABNs. □

5 Optimal Control Problem

For the logical dynamical system (10), (11) expressing the Petri net (8), consider the following optimal control problem.

Problem 1. For the logical dynamical system (10), (11) expressing the Petri net (8), suppose that the initial state $x(0) = x_0$ satisfying Assumption 1 is given. Then find an input sequence $v(0), v(1), \ldots, v(N-1)$ minimizing the linear cost function

$$J = \sum_{k=0}^{N-1} \{Qx(k) + Rv(k)\} + Q_f x(N) \tag{19}$$

where $Q, Q_f \in \mathcal{R}^{1\times n_d}$, $R \in \mathcal{R}^{1\times m_d}$ are weighting vectors whose element is a non-negative real number.

For simplicity of discussion, a linear function with respect to x and u is considered as a cost function, but a quadratic cost function may be used. In addition, using the offset vector $x_d \in \{0,1\}^{n_d}$ and $v_d \in \{0,1\}^{m_d}$, $x(k)$ and $v(k)$ may be replaced to $\hat{x}(k) := x(k) - x_d$ and $\hat{v}(k) := v(k) - v_d$. Then it is necessary that the cost function (19) is also replaced to $J = \sum_{i=0}^{N-1} \{Q|\hat{x}(i)| + R|\hat{v}(i)|\} + Q_f|\hat{x}(N)|$. In addition, N must be determined according to a given biological network. Although a longer N is desirable, the computation time to solve Problem 1 must be also considered. For a small N, Problem 1 may be repeatedly solved at each time. This policy is well known as model predictive control [6].

In Problem 1, we assume that an input sequence $v(0), v(1), \ldots, v(N-1)$ is arbitrarily determined. However, there is a possibility that a given biological system does not satisfy this assumption. Then suppose that some candidates of input sequences are given. In Problem 1, the optimal input sequence minimizing the cost function (19) is selected among the set of the candidates $\mathcal{B} \subseteq \{0,1\}^{m_d N}$. This extension is easy. In this sense, Problem 1 can be applied to optimal control of asynchronous Boolean networks such that the updating time of each state is given in advance [12]. Of course, this problem can also be applied to optimal control of SBNs. Thus Problem 1 includes several situations.

Next, we show an example for setting weighting vectors from the biological viewpoint.

Example 4. Consider the Boolean network expressing an apoptosis network in Example 1 again. From (4), we obtain the Petri net (8) with 6 places, 2 external input places, and 12 transitions. In addition, from the obtained Petri net, we obtain the logical dynamical system (10), (11). For the obtained logical dynamical system, we consider to find a control strategy such that a stimulus is not applied as much as possible, and cell survival is achieved. $u(k) = 0$ implies that a stimulus is not applied to the system, and $x_1(k) = 1$, $x_2(k) = 0$ express cell survival [9]. Then as one of appropriate cost functions, we can consider the following cost function

$$J = \sum_{k=0}^{N-1} \{10|x_1(k) - 1| + 10|x_2(k) - 0| + u(k)\}$$

$$+100|x_1(N)-1|+100|x_2(N)-0|.$$

By the appropriate coordinate transformation, this cost function can be rewritten as the form of (19). See also [10–12] for biological examples on the optimal control problems. □

6 Reduction to an Integer Linear Programming Problem

Finally, let us consider to reduce Problem 1 to an integer linear programming (ILP) problem.

Problem 1 can be rewritten by using (10), (11). First, by using

$$x(k) = A^k x_0 + \sum_{i=1}^{k} A^{i-1} B v(k-i)$$

obtained from the state equation (10), we obtain

$$\bar{x} = \bar{A} x_0 + \bar{B} \bar{v} \tag{20}$$

where $\bar{x} := [\ x^T(0)\ \ x^T(1)\ \ \cdots\ \ x^T(N)\ ]^T$, $\bar{v} := [\ v^T(0)\ \ v^T(1)\ \ \cdots\ \ v^T(N-1)\ ]^T$ and

$$\bar{A} := \begin{bmatrix} I_{n_d} \\ A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, \quad \bar{B} := \begin{bmatrix} 0_{n_d \times m_d} & 0 & \cdots & 0 \\ B & 0 & \cdots & 0 \\ AB & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ A^{N-1}B & \cdots & AB & B \end{bmatrix}.$$

Furthermore, from the linear inequality (11), we obtain

$$\bar{C}\bar{x} + \bar{D}\bar{v} \leq \bar{E} \tag{21}$$

where

$$\begin{aligned} \bar{C} &:= [\ \text{block-diag}(C, C, \ldots, C)\ \ 0\ ], \\ \bar{D} &:= \text{block-diag}(D, D, \ldots, D), \\ \bar{E} &:= \left[\ E^T\ \ E^T\ \ \cdots\ \ E^T\ \right]^T. \end{aligned}$$

Next, the cost function (19) can also be rewritten as

$$J = \bar{Q}\bar{x} + \bar{R}\bar{v} \tag{22}$$

where $\bar{Q} := [\ Q\ \ \cdots\ \ Q\ \ Q_f\ ]$ and $\bar{R} := [\ R\ \ \cdots\ \ R\ ]$. By substituting (20) into (21) and (22), we obtain the following theorem.

Theorem 1. *Problem 1 is equivalent to the following ILP problem.*

Problem A:

$$\begin{aligned} \text{find} \quad & \bar{v} \in \{0,1\}^{m_d N}, \\ \min \quad & (\bar{R} + \bar{Q}\bar{B})\bar{v} + \bar{Q}\bar{A}x_0, \\ \text{subject to} \quad & \begin{bmatrix} \bar{C}\bar{B} + \bar{C} \\ -\bar{L}\bar{W} \end{bmatrix} \bar{v} \leq \begin{bmatrix} \bar{E} - \bar{C}\bar{A}x_0 \\ -\ln\,\rho \end{bmatrix}. \end{aligned}$$

Problem A can be solved by using a suitable ILP solver such as IBM ILOG CPLEX Optimizer [24].

7 Conclusion

In this paper, we have discussed optimal control of asynchronous Boolean networks with control inputs. First, we have proposed a method to transform Boolean networks with control inputs into Petri net with external input places. Next, after the obtained Petri net is transformed into a logical dynamical system, the optimal control problem has been formulated. This problem is a general formulation including several biological situations. Finally, the optimal control problem has been reduced to an integer linear programming problem. The proposed approach will be effective for control of several biological systems modeled by Boolean networks.

One of the future works is to apply the proposed approach to biological Boolean networks. From the practical viewpoint, an extension to probabilistic Boolean networks is also important. In addition, for large-scale Boolean networks, the computation time to solve the problem will be long. So it is significant to consider to reduce the computation time to solve the problem. Then one of methods to overcome this difficulty is to use a SAT (satisfiability problem) solver such as zChaff [26]. It is also important to consider approximate solution methods.

This work was supported by Grant-in-Aid for Young Scientists (B) 23760387 and Scientific Research (C) 21500009.

A Linearization of The Product of Binary Variables

The product of binary variables can be linearized by using the following lemma [7].

Lemma 1. *Suppose that binary variables $\delta_j \in \{0,1\}$, $j \in \mathcal{J}$ are given, where $\mathcal{J}$ is some index set. Then $z = \prod_{j\in\mathcal{J}} \delta_j$ is equivalent to the following linear inequalities*

$$\sum_{j\in\mathcal{J}} \delta_j - z \leq |\mathcal{J}| - 1, \quad -\sum_{j\in\mathcal{J}} \delta_j + |\mathcal{J}|z \leq 0$$

where $|\mathcal{J}|$ is the cardinality of $\mathcal{J}$.

References

1. 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, pp. 670–679, 2007.
2. M. Andrecut, S. A. Kauffman, and A. M. Madni, Evidence of Scale-Free Topology in Gene Regulatory Network of Human Tissues, *Int'l Journal of Modern Physics C*, vol. 19, no. 2, pp. 283–290, 2008.
3. S. Azuma, E. Yanagisawa, and J. Imura, Controllability Analysis of Biosystems Based on Piecewise Affine Systems Approach, *IEEE Trans. on Automatic Control*, vol. 53, no. 1, pp. 139–152, 2008.
4. C. Belta, J. Schug, T. Dang, V. Kumar, G. J. Pappas, H. Rubin, and P. Dunlap, Stability and reachability analysis of a hybrid model of luminescence in the marine bacterium *Vibrio fischeri*, *Proc. 40th IEEE Conf. on Decision and Control*, pp. 869–874, 2001.
5. A. Bemporad and M. Morari, Control of systems integrating logic, dynamics, and constraints. *Automatica*, Vol. 35, pp. 407–427, 1999.
6. E. F. Camacho, C. Bordons, *Model Predictive Control*, 2nd Edition, Springer, 2004.
7. T. M. Cavalier, P. M. Pardalos, and A. L. Soyster, Modeling and integer programming techniques applied to propositional calculus, *Computer & Operations Research*, vol. 17, no. 6, pp. 561–570, 1990.
8. C. Chaouiya, E. Remy, P. Ruet, and D. Thieffry, Qualitative Modelling of Genetic Networks: from Logical Regulatory Graphs to Standard Petri Nets, *25th Int'l Conf. on Application and Theory of Petri Nets*, *LNCS 3099*, pp. 137–156, 2004.
9. M. Chaves, Methods for Qualitative Analysis of Genetic Networks, *Proc. European Control Conference 2009*, pp. 671–676, 2009.
10. A. Datta, A. Choudhary, M. L. Bittner, and E. R. Dougherty, External Control in Markovian Genetic Regulatory Networks, *Machine Learning*, vol. 52, pp. 169–191, 2003.
11. A. Datta, A. Choudhary, M. L. Bittner, and E. R. Dougherty, External control in Markovian genetic regulatory networks: the imperfect information case, *Bioinfomatics*, vol. 20, pp. 924–930, 2004.
12. B. Faryabi, J.-F. Chamberland, G. Vahedi, A. Datta, and E. R. Dougherty, Optimal Intervention in Asynchronous Genetic Regulatory Networks, *IEEE Journal of Selected Topics in Signal Processing*, Vol. 2, No. 3, pp. 412–423, 2008.
13. A. Ichikawa and K. Hiraishi, Analysis and Control of Discrete Event Systems Represented by Petri Nets, *Discrete Event Systems: Models and Applications*, *LNCIS 103*, pp. 115–134, 1988.
14. H. D. Jong, Modeling and Simulation of Genetic Regulatory Systems: A Literature Review, *Journal of Computational Biology*, vol. 9, pp. 67–103, 2002.
15. S. A. Kauffman, Metabolic stability and epigenesis in randomly constructed genetic nets, *Journal of Theoretical Biology*, vol. 22, pp. 437–467, 1969.
16. M.-S. Kim *et al.*, DNA demethylation in hormone-induced transcriptional derepression, *Nature*, vol. 461, pp. 1007–1012, 2009.
17. K. Kobayashi, J. Imura, and K. Hiraishi, Polynomial-Time Algorithm for Controllability Test of a Class of Boolean Biological Networks, *EURASIP Journal on Bioinformatics and Systems Biology*, vol. 2010, Article ID 210685, 12 pages, 2010.
18. C. J. Langmead and S. K. Jha, Symbolic Approaches to Finding Control Strategies in Boolean Networks, *Journal of Bioinformatics and Computational Biology*, vol. 7, no. 2, pp. 323–338, 2009.

19. H. Santos-Rosa and C. Caldas, Chromatin modifier enzymes, the histone code and cancer, *European Journal of Cancer*, vol. 41, pp. 2381–2402, 2005.
20. I. Shmulevich, E. R. Dougherty, S. Kim, and W. Zhang, Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks, *Bioinformatics*, vol. 18, pp. 261–274, 2002.
21. L.J. Steggles, R. Banks, and A. Wipat, Modelling and Analysing Genetic Networks: From Boolean Networks to Petri Nets, *Int'l Conf. on Computational Methods in Systems Biology*, *LNBI 4210*, pp. 127–141, 2006.
22. L. Tournier and M.Chaves, Uncovering operational interactions in genetic networks using asynchronous Boolean dynamics, *Journal of Theoretical Biology*, vol. 260, pp. 196–209, 2009.
23. T. Ushio, Maximally Permissive Feedback and Modular Control Synthesis in Petri Nets with External Input Places, *IEEE Trans on Automatic Control*, Vol. 35, No. 7, pp. 844–848, 1990.
24. http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/
25. http://www.informatik.uni-hamburg.de/TGI/PetriNets/
26. http://www.princeton.edu/~chaff/zchaff.html