PAPER Special Section on Mathematical Systems Science and its Applications

A BDD-Based Approach to Finite-Time Control of Boolean Networks

Fuma MOTOYAMA†, *Nonmember*, **Koichi KOBAYASHI**†a), *and* **Yuh YAMASHITA**†, *Members*

SUMMARY Control of complex networks such as gene regulatory networks is one of the fundamental problems in control theory. A Boolean network (BN) is one of the mathematical models in complex networks, and represents the dynamic behavior by Boolean functions. In this paper, a solution method for the finite-time control problem of BNs is proposed using a BDD (binary decision diagram). In this problem, we find all combinations of the initial state and the control input sequence such that a certain control specification is satisfied. The use of BDDs enables us to solve this problem for BNs such that the conventional method cannot be applied. First, after the outline of BNs and BDDs is explained, the problem studied in this paper is given. Next, a solution method using BDDs is proposed. Finally, a numerical example on a 67-node BN is presented.

key words: *binary decision diagram (BDD), Boolean networks, finite-time control, gene regulatory networks*

1. Introduction

A Boolean network (BN) is well known as one of the mathematical models in complex networks such as gene regulatory networks. In a BN, the state and the control input take a binary value, and time evolution of the state is represented by a set of Boolean functions. There is a weakness that a BN is too simple, but a BN is widely used as the first step of developing control theory of complex networks. Many fundamental results have been obtained, such as stability [1], [2], stabilization [3], [4], controllability [5]–[7], observability [5], [8], [9], and optimal control [10]–[12]. To model more complex behavior, a probabilistic Boolean network (PBN) [13] and context-sensitive PBN [14] have been proposed as an extended model of BNs. For also these extended models, fundamental results [15]–[18] have been obtained.

In the last decade, the semi-tensor product (STP) method has been widely used in analysis and control of BNs (see, e.g., [5] and [19]). Using the STP method, a given BN is equivalently transformed into a linear algebraic representation. Hence, analysis/control problems can be solved in an algebraic way. However, the STP method has a weakness. For a BN with n nodes and m control inputs, matrices of the size $2^n \times 2^{n+m}$ must be handled. From this fact, BNs such that the STP method can be applied are limited.

A BDD (binary decision diagram) and SAT (satisfiability)/SMT (satisfiability modulo theories) solvers are well known as efficient tool for handling Boolean functions. A BDD is a data structure used to represent Boolean functions [20]. An efficient algorithm for performing logical operations (AND, OR, XOR, and so on) on BDDs has been proposed. A BDD efficiently works analysis of BNs, such as attractor detection [21], [22]. SAT/SMT solvers determine whether a given Boolean function is satisfiable or not. The Yices SMT solver (https://yices.csl.sri.com/), the Z3 Theorem Prover (https://github.com/Z3Prover) and so on have been developed. SAT/SMT solvers have been used for attractor detection [23] and design of BN based on attractors [24]. To the best of our knowledge, BDDs and SAT/SMT solvers have not been applied to the control problem of BNs.

In this paper, using BDDs, we propose a solution method for the finite-time control problem of BNs. The finite-time control problem is to find a control input sequence such that the state at a given final time is equal to the target value under a given initial state. This problem is one of the typical control problems in BNs [25]. Using the STP method, this problem has been studied as also the controllability problem. However, a class of BNs is limited due to the above reason. Furthermore, in [25], it has been proven that the finite-time control problem for a general BN is NP-hard. In such cases, it is important to utilize efficient computation tools such as BDDs and SAT/SMT solvers. Since Boolean functions are simplified by using BDDs, it is appropriate to use BDDs for this problem. In this paper, using BDDs, we consider finding all combinations of the initial state and the control input sequence such that a certain control specification is satisfied (i.e., the initial state is not given in advance). Hence, the problem studied in this paper is a more general problem including the problem in [25] as a special case.

This paper is organized as follows. In Sect. 2, BNs and BDDs are summarized. In Sect. 3, the problem studied here is formulated. In Sect. 4, a solution method using BDDs is proposed. A simple example is also presented to demonstrate the proposed method. In Sect. 5, the proposed method is applied to a 67-node BN. The STP method cannot be applied to such BNs. We compare the computation time using BDDs with that using the Z3 Theorem Prover. In Sect. 6, we conclude this paper, and address future efforts.

Notation: Let $\{0,1\}^n$ denote the set of n-dimensional vectors, which consists of elements 0 and 1. Let 0_n (1_n) denote the n-dimensional vector whose elements are all 0 (1).

Manuscript received April 16, 2023.
Manuscript revised August 23, 2023.
Manuscript publicized October 23, 2023.

†The authors are with the Graduate School of Information Science and Technology, Hokkaido University, Sapporo-shi, 060-0814 Japan.
a) E-mail: k-kobaya@ssi.ist.hokudai.ac.jp

DOI: 10.1587/transfun.2023MAP0006

2. Preliminaries

2.1 Boolean Networks

A BN consists of a set of n nodes $V = \{x_1, x_2, \ldots, x_n\}$ and a set of n Boolean functions $F_a = \{f_1^a, f_2^a, \ldots, f_n^a\}$. In the case of gene regulatory networks, V and F_a correspond to a set of genes and a set of gene regulatory rules, respectively. For the i-th node, a Boolean variable $x_i \in \{0, 1\}$ (e.g., expression level of the gene) and a Boolean function f_i^a are associated. Each Boolean function is given based on interactions between nodes (i.e., the network structure), where logical operators such as logical AND ($\wedge$), logical OR ($\vee$), and logical NOT ($\neg$) are used. Then, we can obtain the following expression:

$$\begin{cases} x_1(t+1) = f_1^a(x(t)), \\ x_2(t+1) = f_2^a(x(t)), \\ \vdots \\ x_n(t+1) = f_n^a(x(t)), \end{cases} \tag{1}$$

where $x = [x_1, x_2, \ldots, x_n] \in \{0, 1\}^n$ is the state, and $t = 0, 1, 2, \ldots$ is the discrete time.

Example 1: Consider the following BN model for a simplified apoptosis network [26]:

$$\begin{cases} x_1(t+1) = x_1(t), \\ x_2(t+1) = x_1(t) \wedge \neg x_3(t), \\ x_3(t+1) = \neg x_2(t) \wedge x_4(t), \\ x_4(t+1) = x_1(t) \vee x_3(t), \end{cases}$$

where x_1 is the concentration level (high or low) of the tumor necrosis factor (TNF, a stimulus), x_2 is the concentration level of the inhibitor of apoptosis proteins (IAP), x_3 is the concentration level of the active caspase 3 (C3a), and x_4 is the concentration level of the active caspase 8 (C8a). If $x(t) = [1, 1, 0, 0]$, the next time state is $x(t+1) = [1, 1, 0, 1]$. See, e.g., [27] for more complicated models of an apoptosis network. □

2.2 Binary Decision Diagrams

A BDD is a data structure that efficiently represents a Boolean function. A BDD consists of two terminal nodes labeled with 0 and 1, and nodes labeled with variable names. Each node except for the terminal nodes has two child nodes, and one child node is chosen according to the value of the (parent) node. When the terminal node is reached by continuing to choose child nodes according to each value of node, the value of terminal node is the output of the function. It is known that a BDD is uniquely determined from a Boolean function when the order of variables on the graph is fixed.

The Apply operation has been proposed as a useful algorithm for handling logical equations on BDDs [20]. Apply operations can perform logical operators (AND, OR, XOR, etc.) between BDDs. The computation time of the Apply operation is roughly proportional to the total amount of input and output data. Figure 1 shows the procedure to generate the logical expression $a \wedge b \vee \neg c$ by Apply operation.

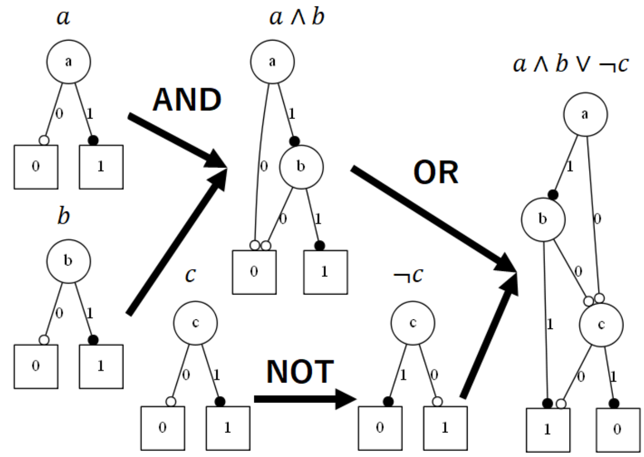


Fig. 1 The procedure to generate the logical expression $a \wedge b \vee \neg c$ by Apply operation.

3. Problem Formulation

Consider the following BN that is added the control input to the BN (1):

$$\begin{cases} x_1(t+1) = f_1(x(t), u(t)), \\ x_2(t+1) = f_2(x(t), u(t)), \\ \vdots \\ x_n(t+1) = f_n(x(t), u(t)), \end{cases} \tag{2}$$

where $u = [u_1, u_2, \ldots, u_m] \in \{0, 1\}^m$ is the control input and f_i is a given Boolean function. We assume that for each element of the control input, any binary value can be set at each discrete time. Consider the following finite-time control problem for the BN (2).

Problem 1: Suppose that for the BN (2), the target state $x^* \in \{0, 1\}^n$ and the final time T are given.

i) Find all initial states such that $x(T) = x^* = [x_1^*, x_2^*, \ldots, x_n^*]$ holds. Let $\mathcal{X}_0 \subseteq \{0, 1\}^n$ denote the obtained initial state set.
ii) For each $x_0 \in \mathcal{X}_0$, find a control input sequence $U = [u(0), u(1), \ldots, u(T-1)]$.

In the case where the initial state is given, this problem is one of the typical control problems for BNs, and has been studied in [25].

4. Proposed Solution Method

In this section, we propose a solution method for Problem 1. First, details of the proposed solution method is explained. Next, the proposed solution method is demonstrated by a simple example.

4.1 Solution Method Using BDDs

Consider solving Problem 1 using BDDs.

First, defining $f := [f_1, f_2, \ldots, f_n]$, the final state $x(T)$ can be represented by

$$\begin{aligned} x(T) &= f(f(\cdots f(f(x(0), u(0)), u(1)), \ldots), u(T-1)) \\ &=: f^{(T)}(x(0), U), \end{aligned}$$

where the Boolean function $f^{(T)} : \{0,1\}^n \times \{0,1\}^{Tm}$ can be calculated from f. In addition, $f^{(T)}$ is represented as $f^{(T)} = [f_1^{(T)}, f_2^{(T)}, \ldots, f_n^{(T)}]$. Then, Problem 1 can be rewritten as the following problem.

Problem 2: Find all combinations of the pair of the initial state $x(0)$ and the control input sequence U such that the following logical formula $F(x(0), U)$ is equal to 1 (true):

$$\begin{aligned} F(x(0), U) := &(f_1^{(T)}(x(0), U) \leftrightarrow x_1^*) \\ &\wedge (f_2^{(T)}(x(0), U) \leftrightarrow x_2^*) \\ &\wedge \cdots \wedge (f_n^{(T)}(x(0), U) \leftrightarrow x_n^*), \end{aligned} \tag{3}$$

where "$\leftrightarrow$" represents the logical equivalence operator.

To solve this problem, $F(x(0), U)$ is represented by a single BDD. Then, a solution to this problem can be easily derived from the obtained BDD.

Finally, we summarize the proposed procedure for solving Problem 2, which can be implemented on BDDs.

Procedure of solving Problem 2:

Step 1: Derive a BDD representing the Boolean function $f_i^{(T)}(x(0), U)$, $i = 1, 2, \ldots, n$ by recursively substituting Boolean functions into other one.

Step 2: Derive a BDD representing $F(x(0), U)$ of (3).

Step 3: Find all combinations of $x(0)$ and U such that $F(x(0), U) = 1$.

Since Problem 2 is called an AllSAT (all solutions satisfiability) problem, it may be solved using a SAT/SMT (Satisfiability Modulo Theories) solver. However, using BDD, we can obtain a compact representation, which will be useful in analysis and control of BNs[†].

4.2 Example

We present a simple example to explain the proposed procedure for solving Problem 2. Although the proposed procedure can be implemented on BDDs, we explain here the procedure using logical formulas.

Consider the following BN with three nodes and a single control input:

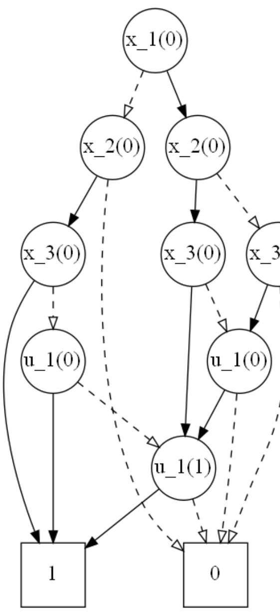


Fig. 2 BDD of $F(x(0), U)$ in the example in Sect. 4.2. The nodes x_1(0), x_2(0), and x_3(0) correspond to $x_1(0)$, $x_2(0)$, and $x_3(0)$, respectively. The nodes u_1(0) and u_1(1) correspond to $u_1(0)$ and $u_1(1)$, respectively.

$$\begin{cases} x_1(t+1) = (x_1(t) \wedge x_3(t) \wedge u_1(t)) \vee x_2(t), \\ x_2(t+1) = x_3(t) \vee u_1(t), \\ x_3(t+1) = \neg x_1(t). \end{cases} \tag{4}$$

Suppose that the target state x^* and the final time T are given by $x^* = [1, 1, 0]$ and $T = 2$, respectively. Consider finding all combinations of the initial states and the control sequences such that $x(2) = [1, 1, 0]$ holds.

In Step 1, the Boolean function $f_i^{(2)}$ can be derived as follows. From $x_1(2) = (x_1(1) \wedge x_3(1) \wedge u_1(1)) \vee x_2(1)$ and (4), we can obtain

$$f_1^{(2)} = (\neg x_1(0) \wedge x_2(0) \wedge u_1(1)) \vee x_3(0) \vee u_1(0).$$

In a similar way, from $x_2(2) = x_3(1) \vee u_1(1)$, $x_3(2) = \neg x_1(1)$, and (4), the Boolean functions $f_2^{(2)}$ and $f_3^{(2)}$ can be obtained as

$$\begin{aligned} f_2^{(2)} &= \neg x_1(0) \vee u_1(1), \\ f_3^{(2)} &= \neg(x_1(0) \wedge x_3(0) \wedge u_1(0)) \wedge \neg x_2(0), \end{aligned}$$

respectively.

In Step 2, the Boolean function $F(x(0), U)$ is obtained as follows:

$$\begin{aligned} F(x(0), U) &= (f_1^{(2)} \leftrightarrow x_1^*) \wedge (f_2^{(2)} \leftrightarrow x_2^*) \wedge (f_3^{(2)} \leftrightarrow x_3^*) \\ &= \{(\neg x_1(0) \wedge x_2(0) \wedge u_1(1)) \vee x_3(0) \vee u_1(0)\} \\ &\quad \wedge (\neg x_1(0) \vee u_1(1)) \\ &\quad \wedge \neg\{\neg(x_1(0) \wedge x_3(0) \wedge u_1(0)) \wedge \neg x_2(0)\} \\ &= (x_1(0) \wedge x_3(0) \wedge u_1(0) \wedge u_1(1)) \\ &\quad \vee (\neg x_1(0) \wedge x_2(0) \wedge x_3(0)) \\ &\quad \vee (\neg x_1(0) \wedge x_2(0) \wedge u_1(0)) \end{aligned}$$

[†]A BDD-based SMT solver has been developed (see, e.g., [28]). In this paper, we consider using only BDD techniques.

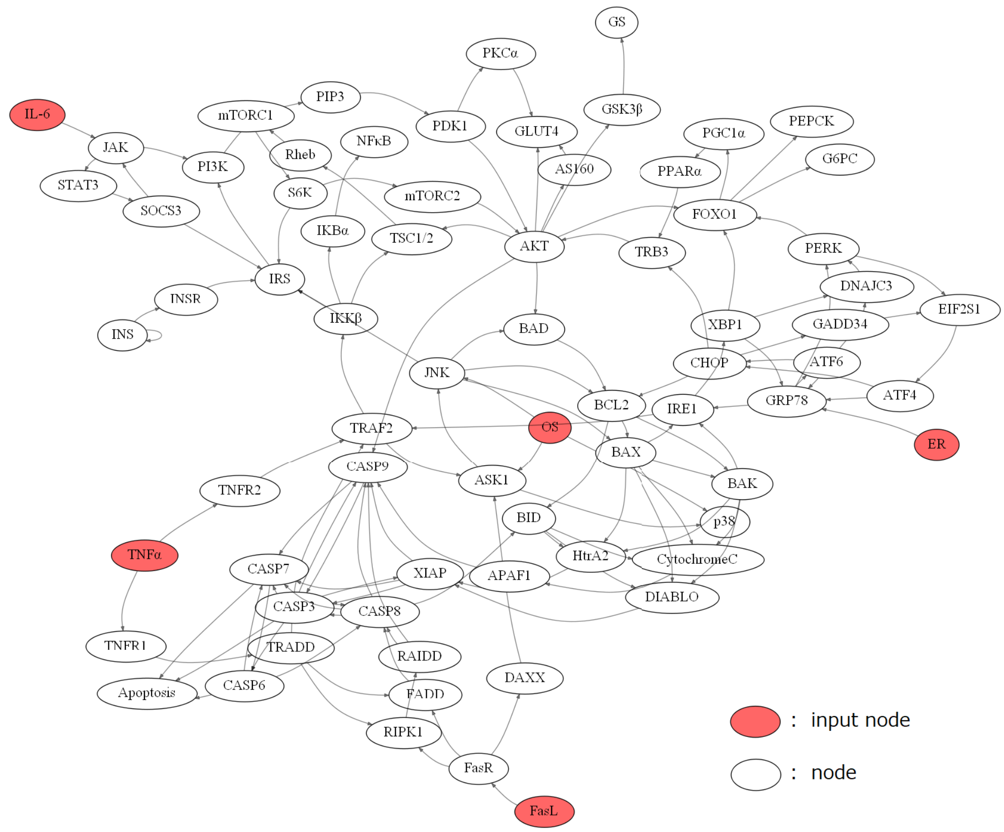


Fig. 3 Graph representing the interactions between nodes, where input nodes correspond to elements of the control input.

$$\begin{aligned}&\vee(\neg x_1(0)\wedge x_2(0)\wedge u_1(1))\\&\vee(x_2(0)\wedge x_3(0)\wedge u_1(1))\\&\vee(x_2(0)\wedge u_1(0)\wedge u_1(1)).\end{aligned}$$

The BDD representing $F(x(0), U)$ is shown in Fig. 2.

In Step 3, all combinations of the initial state and the control sequence that satisfy $F(x(0), U) = 1$ are found from Fig. 2. In this graph, the combinations of nodes that reach the terminal node 1 satisfies $F(x(0), U) = 1$. From observation of this graph, the combinations are obtained as

$$\begin{aligned}&[x_1(0), x_2(0), x_3(0), u_1(0), u_1(1)]\\&=[0,1,0,0,1],[0,1,0,1,*],[0,1,1,*,*],\\&\quad[1,0,1,1,1],[1,1,0,1,1],[1,1,1,*,1],\end{aligned}$$

where "$*$" means that the value can be given by any binary value. For example, $[0,1,1,*,*]$ means that when $x_0 = [0,1,1]$, the control input sequence can be taken any binary value (i.e., $[0,0],[0,1],[1,0],[1,1]$). Therefore, we see that there are 11 combinations. Thus, we can derive the solution of Problem 1.

5. Numerical Example

In this section, to demonstrate the proposed method, we present a numerical example of a BN such that the STP method cannot be applied.

Consider a 67-node, 5-input BN model of an apoptotic network [29]. The interactions between nodes are shown in Fig. 3, where input nodes correspond to elements of the control input. Suppose that the target state $x^* \in \{0,1\}^{67}$ is given as the following binary vector:

$$x^* = [0_{16}, 1, 0_{16}, 1, 0_3, 1_{10}, 0, 1, 0_7, 1_3, 0_7, 1]\,.$$

We consider the cases of $T = 1, 2, \ldots, 15$. We remark here that the STP method, which is frequently used in analysis and control of BNs, cannot be applied to such BNs. This is because in the STP method, matrices with $2^{67} \times 2^{67+5}$ must be generally handled, and handling such matrices is hard on a conventional computer. See, e.g., [5] and [19] for details of the STP method. In this numerical example, we compare the computation time of the proposed procedure with that of the SMT solver.

We present the computation result. Table 1 shows the number of combinations of $x(0)$ and U, the computation time of the proposed procedure, and that of the SMT solver, where we use Python 3.9.7 on the computer (CPU: AMD Ryzen 5 5600X 6-Core Processor 3.70 GHz, Memory: 32.0 GB, OS: Windows 11). We also use z3-solver 4.12.2.0 as the SMT solver. We set the timeout to 10000 sec. From Table 1, we see that the computation time of the proposed procedure is faster than that of the SMT solver.

Table. 1 The computation result and the computation time for each T.

T	The number of combinations of $x(0)$ and U that reach x^*	Computation time of the proposed method (sec.)	Computation time of the SMT solver (sec.)
1	49152	0.00	0.09
2	27525120	0.00	0.15
3	812646400	0.02	1.39
4	26817331200	0.04	4.41
5	236165529600	0.20	31.08
6	920466227200	0.32	74.45
7	2460580577280	1.43	315.85
8	11803998289920	3.74	503.08
9	60487995752448	10.86	5792.54
10	229649820942336	23.16	10000 <
11	1089591461281792	49.62	10000 <
12	4360614260506624	83.34	10000 <
13	16290233280626688	120.56	10000 <
14	48994655819268096	167.57	10000 <
15	147099776955973632	227.29	10000 <

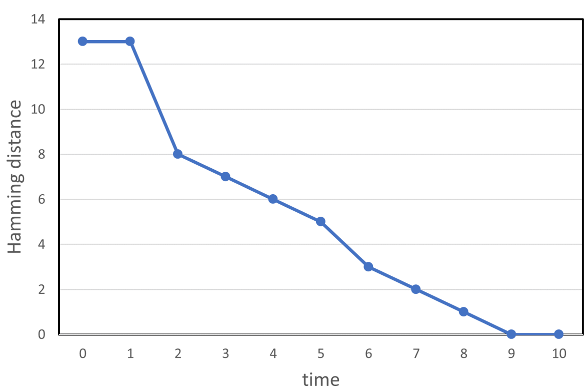


Fig. 4 Time response of the Hamming distance.

We present one sample of combinations in the case of $T = 10$. As one of combinations, we can obtain the following initial state and control input sequence:

$$
\begin{aligned}
x(0) = [&0_5, 1, 0_{10}, 1, 0_3, 1, 0_{12}, 1, 0_3, 1_3, 0, 1, 0_2, 1_2,\\
&0_3, 1_3, 0_3, 1_5, 0_2, 1, 0_3, 1], \qquad (5)\\
u(0) &= [1, 0, 0, 0, 1],\\
u(1) &= u(2) = [1, 0, 0, 0, 0],\\
u(3) &= [0, 0, 0, 0, 0],\\
u(4) &= [0, 0, 0, 0, 1],\\
u(5) &= [0, 0, 0, 0, 0],\\
u(6) &= [1, 0, 0, 0, 0],\\
u(7) &= [0, 0, 0, 0, 0],\\
u(8) &= [1, 0, 0, 0, 0],\\
u(9) &= [0, 0, 0, 0, 0],
\end{aligned}
$$

Instead of time response of the state, we present the Hamming distance defined by $\sum_{i=1}^{67} |x_i(k) - x_i^*(k)|$. Figure 4 shows the Hamming distance at each time. From this figure, we see that $x(T) = x^*$ holds.

Finally, we further discuss the case where the initial state is given by (5). In this case, there are other control input sequences such that $x(T) = x^*$ holds. All control input sequences such that $x(T) = x^*$ holds can be characterized as follows:

$$
\begin{aligned}
u(0) &= [*, 0, 0, 0, 1],\\
u(1) &= [*, 0, 0, 0, *].\\
u(2) &= [0, 0, 0, 0, 0],\\
u(3) &= [*, 0, 0, 0, *],\\
u(4) &= [*, 0, 0, 0, 1],\\
u(5) &= u(6) = [*, 0, 0, 0, 0],\\
u(7) &= [*, 0, 0, 0, *],\\
u(8) &= u(9) = [*, 0, 0, 0, 0].
\end{aligned}
$$

Hence, there are many patterns of the control input sequence such that $x(T) = x^*$ holds. For example, when it is desirable that the control input is zero as well as possible, we may set that $u_5(0)$ and $u_5(4)$ are equal to 1, and other inputs are zero. Thus, by the proposed method, we can obtain useful information.

6. Conclusion

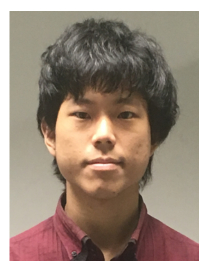
In this paper, based on BDDs, we proposed a solution method for the finite-time control problem of BNs. Using the proposed method, we can obtain all combinations of the initial state and the control input sequence such that the state at the final time reaches the target state. The effectiveness of the proposed method was presented by a BN with 67-node and 5-input.

In future work, using the BDD obtained by the proposed method, we will consider developing a method for deriving both the initial state and the control input sequence such that an other control specification is satisfied. Moreover, as an extension of the proposed method, it is also significant to develop a solution method for the optimal control problem using BDDs.

This work was partly supported by JSPS KAKENHI Grant Numbers JP21H04558, JP22K04163, JP23H01430.

References

[1] S. Azuma, T. Yoshida, and T. Sugie, "Structural monostability of activation-inhibition Boolean networks," IEEE Trans. Control Netw. Syst., vol.4, no.2, pp.179–190, 2017.
[2] M. Meng, J. Lam, J.E. Feng, and K.C. Cheung, "Stability and stabilization of Boolean networks with stochastic delays," IEEE Trans. Autom. Control, vol.64, no.2, pp.790–796, 2018.
[3] Y. Guo, P. Wang, W. Gui, and C. Yang, "Set stability and set stabilization of Boolean control networks based on invariant subsets," Automatica, vol.61, pp.106–112, 2015.
[4] A. Yerudkar, C. Del Vecchio, and L. Glielmo, "Feedback stabilization control design for switched Boolean control networks," Automatica, vol.116, p.108934, 2020.
[5] D. Cheng and H. Qi, "Controllability and observability of Boolean control networks," Automatica, vol.45, no.7, pp.1659–1667, 2009.
[6] D. Laschov and M. Margaliot, "Controllability of Boolean control networks via the perron–frobenius theory," Automatica, vol.48, no.6, pp.1218–1223, 2012.
[7] Q. Zhu, Y. Liu, J. Lu, and J. Cao, "Further results on the controllability of Boolean control networks," IEEE Trans. Autom. Control, vol.64, no.1, pp.440–442, 2018.
[8] E. Fornasini and M.E. Valcher, "Observability, reconstructibility and state observers of Boolean control networks," IEEE Trans. Autom. Control, vol.58, no.6, pp.1390–1401, 2013.
[9] Y. Yu, M. Meng, J.e. Feng, and G. Chen, "Observability criteria for Boolean networks," IEEE Trans. Autom. Control, vol.67, no.11, pp.6248–6254, 2022.
[10] K. Kobayashi and K. Hiraishi, "Optimal control of gene regulatory networks with effectiveness of multiple drugs: A Boolean network approach," BioMed research international, vol.2013, 2013.
[11] E. Fornasini and M.E. Valcher, "Optimal control of Boolean control networks," IEEE Trans. Autom. Control, vol.59, no.5, pp.1258–1270, 2014.
[12] Y. Wu, X.M. Sun, X. Zhao, and T. Shen, "Optimal control of Boolean control networks with average cost: A policy iteration approach," Automatica, vol.100, pp.378–387, 2019.
[13] 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, no.2, pp.261–274, 2002.
[14] R. Pal, A. Datta, M.L. Bittner, and E.R. Dougherty, "Intervention in context-sensitive probabilistic Boolean networks," Bioinformatics, vol.21, no.7, pp.1211–1218, 2005.
[15] F. Li and J. Sun, "Controllability of probabilistic Boolean control networks," Automatica, vol.47, no.12, pp.2765–2771, 2011.
[16] K. Kobayashi and K. Hiraishi, "An integer programming approach to optimal control problems in context-sensitive probabilistic Boolean networks," Automatica, vol.47, no.6, pp.1260–1264, 2011.
[17] M. Toyoda and Y. Wu, "Mayer-type optimal control of probabilistic Boolean control network with uncertain selection probabilities," IEEE Trans. Cybern., vol.51, pp.3079–3092, 2021.
[18] A. Acernese, A. Yerudkar, L. Glielmo, and C. Del Vecchio, "Reinforcement learning approach to feedback stabilization problem of probabilistic Boolean control networks," IEEE Control Syst. Lett., vol.5, no.1, pp.337–342, 2020.
[19] D. Cheng, H. Qi, and Z. Li, Analysis and Control of Boolean Networks: A Semi-Tensor Product Approach, Springer, London, U.K., 2011.
[20] R.E. Bryant, "Graph-based algorithms for Boolean function manipulation," IEEE Trans. Comput., vol.C-35, no.8, pp.677–691, 1986.
[21] G.V. Trinh, T. Akutsu, and K. Hiraishi, "An FVS-based approach to attractor detection in asynchronous random Boolean networks," IEEE/ACM Trans. Comput. Biol. Bioinf., vol.19, no.2, pp.806–818, 2022.
[22] Q. Yuan, H. Qu, J. Pang, and A. Mizera, "Improving BDD-based attractor detection for synchronous Boolean networks," Sci. China Inf. Sci., vol.59, no.8, pp.1–16, 2016.
[23] T. Tamura and T. Akutsu, "Detecting a singleton attractor in a Boolean network utilizing SAT algorithms," IEICE Trans Fundamentals, vol.E92-A, no.2, pp.493–501, Feb. 2009.
[24] K. Kobayashi and K. Hiraishi, "ILP/SMT-based method for design of Boolean networks based on singleton attractors," IEEE/ACM Trans. Comput. Biol. Bioinf., vol.11, no.6, pp.1253–1259, 2014.
[25] 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, pp.670–679, 2007.
[26] M. Chaves, "Methods for qualitative analysis of genetic networks," 2009 European Control Conference, pp.671–676, 2009.
[27] L. Tournier and M. Chaves, "Uncovering operational interactions in genetic networks using asynchronous Boolean dynamics," Journal of Theoretical Biology, vol.260, no.2, pp.196–209, 2009.
[28] M. Jonáš and J. Strejček, "Q3B: An efficient BDD-based SMT solver for quantified bit-vectors," International Conference on Computer Aided Verification, pp.64–73, Springer, 2019.
[29] P. Dutta, L. Ma, Y. Ali, P. Sloot, and J. Zheng, "Boolean network modeling of β-cell apoptosis and insulin resistance in type 2 diabetes mellitus," BMC Syst. Biol., vol.13, no.2, pp.1–12, 2019.



Fuma Motoyama received the B.E. degree in 2020 and the M.I.S degree in 2022 from Hokkaido University. He is currently a doctor course student at the Graduate School of Information Science and Technology, Hokkaido University. His research interests include Boolean networks.

Koichi Kobayashi received the B.E. degree in 1998 and the M.E. degree in 2000 from Hosei University, and the D.E. degree in 2007 from Tokyo Institute of Technology. From 2000 to 2004, he worked at Nippon Steel Corporation. From 2007 to 2015, he was an Assistant Professor at Japan Advanced Institute of Science and Technology. From 2015 to 2022, he was an Associate Professor at Hokkaido University. Since 2023, he has been a Professor at the Faculty of Information Science and Technology, Hokkaido University. His research interests include discrete event and hybrid systems. He is a member of IEEE, IEEJ, IEICE, ISCIE, and SICE.

Yuh Yamashita received his B.E., M.E., and Ph.D. degrees from Hokkaido University, Japan, in 1984, 1986, and 1993, respectively. In 1988, he joined the faculty of Hokkaido University. From 1996 to 2004, he was an Associate Professor at the Nara Institute of Science and Technology, Japan. Since 2004, he has been a Professor of the Graduate School of Information Science and Technology, at Hokkaido University. His research interests include nonlinear control and nonlinear dynamical systems. He is a member of SICE, ISCIE, IEICE, RSJ, and IEEE.