On Attractor Detection and Optimal Control of Deterministic Generalized Asynchronous Random Boolean Networks

Trinh Van Giang and Kunihiko Hiraishi, *Member, IEEE*

Abstract—Deterministic asynchronous Boolean networks play a crucial role in modeling and analysis of gene regulatory networks. In this paper, we focus on a typical type of deterministic asynchronous Boolean networks called deterministic generalized asynchronous random Boolean networks (DGARBNs). We first formulate the extended state transition graph, which captures the whole dynamics of a DGARBN and paves potential ways to analyze this DGARBN. We then propose two SMT-based methods for attractor detection and optimal control of DGARBNs. These methods are implemented in a JAVA tool called DABoolNet. Two experiments are designed to highlight the scalability of the proposed methods. We also formally state and prove several relations between DGARBNs and other models including deterministic asynchronous models, block-sequential Boolean networks, generalized asynchronous random Boolean networks, and mixed-context random Boolean networks. Several case studies are presented to show the applications of our methods.

Keywords—gene regulatory network, deterministic asynchronous Boolean model, attractor, optimal control, SMT

1 Introduction

Boolean networks (BNs) play a crucial role in modeling and analysis of complex biological networks (e.g., gene regulatory networks [1]). They have also been applied to many areas beyond systems biology, such as, mathematics, neural networks, social modeling, and robotics (see [2]). Many different types of updating schemes of BNs (e.g., synchronous, asynchronous, deterministic, or non-deterministic), which regulate the way that the nodes are updated through time evolution, have been proposed and widely studied. The results of [3], [4], [5], [6] show that different types of updating schemes produce different behaviors of the same BN.

In the biological context, the synchronous updating scheme (e.g., classical random Boolean networks – CRBNs [1]) was criticized because of the assumption that the dynamics of gene regulatory networks (GRNs) is deterministic and synchronous, i.e., all genes change their expression levels simultaneously [7]. Thus, the asynchronous updating scheme, in which all genes take different time to change their expression levels, is closer to biological phenomena [8], [9], [7]. For example, a very recent work [10] has explicitly backed up the necessity of asynchronous models for modelling GRNs over a realistic proof-of-concept case study. The proposed alternative is the non-deterministic asynchronous updating scheme (e.g., asynchronous random Boolean networks – ARBNs [8]) with the assumption that only one randomly selected gene can be updated at a single step. However, it did not give encouraging results, since the networks change drastically their properties due to the non-determinism [11]. Moreover, this updating scheme has also some disadvantages including the high complexity of the state transition graph and the inclusion of many incompatible or unrealistic pathways [12]. With this motivation, a new type of updating, asynchronous but deterministic, was proposed [4], [5]. The deterministic asynchronous updating scheme can help us to model and analyze biological networks more reasonably [11], [13]. For example, we can model asynchronous phenomena which are not random, a thing which is quite difficult with the non-deterministic asynchronous updating scheme [11]. Furthermore, models with deterministic asynchronous updating scheme are particularly useful when information about the kinetics of biological processes is known [14].

To date, there are various types of BNs with the deterministic asynchronous updating scheme: deterministic generalized asynchronous random Boolean networks (DGARBNs) [5], deterministic asynchronous random Boolean networks (DARBNs) [5], mixed-context random Boolean networks (MxRBNs) [11], deterministic asynchronous (DA) models [15], and block-sequential Boolean networks (BSBNs) [4]. They have been widely used in modeling and analysis of gene regulatory networks [15], [13], [16], [17]. In this paper, we focus on DGARBNs which is a typical type of deterministic asynchronous BNs. There are three reasons for this choice.

Firstly, DGARBNs offer an interesting compromise between CRBNs and ARBNs, which could provide a suitable modeling formalism of various types of systems [18]. For example, the authors of [19] applied the updating schemes of DGARBNs and ARBNs to the model of the Spatial Prisoner's Dilemma game which is the most used game in the area of evolutionary game theory. Based on simulations, they obtained that these two updating schemes lead basically the same outcome of the model. Recently, Martin Schneiter et al. [20] used BNs to formulate a simplified pluricellular epithelium model, which intends to present plausibly the self-organization of ciliary beating patterns as well as of the associated fluid transport across the airway epithelium. The simulation results show that DGARBNs lead to more realistic dynamics (flexibility and robustness) and may therefore be favored by evolution.

Secondly, DGARBNs are general and interesting mathematical objects since there is no restriction on their Boolean functions and contexts. For example, CRBNs are a special case of DGARBNs [5], [18]. Moreover, studying DGARBNs can be a good starting point for further studies on more complex networks such as DARBNs or MxRBNs [11]. This is reasonable since both DARBNs and MxRBNs are constructed based on DGARBNs and seem to be more computationally complex than DGARBNs [11], [2].

• *T. Van Giang is with the School of Information Science, Japan Advanced Institute of Science and Technology, Nomi 923-1292, Japan (e-mail: giang-trinh@jaist.ac.jp).*
K. Hiraishi is with the School of Information Science, Japan Advanced Institute of Science and Technology, Nomi 923-1292, Japan (e-mail: hira@jaist.ac.jp).

Manuscript received April 19, 2005; revised January 11, 2007.

Lastly, to our best knowledge, all the existing studies for DGARBNs are theoretical or simulation-based. For example, Carlos Gershenson firstly proposed DGARBNs [5] and analyzed their dynamics by simulations [5], [11]. Li et al. [21] proposed a semi-tensor product-based framework to study many variants of asynchronous BNs including DGARBNs. This approach is deeply theoretical but not practical since the sizes of matrices are exponential with respect to the number of nodes. Thus, analytical and practical studies for DGARBNs are needed. In general, the deterministic behaviors of DGARBNs make them relatively easy to analyze as compared to non-deterministic asynchronous BNs (e.g., ARBNs) [5]. Since many analytical and practical studies have been done for ARBNs [7], [14], such studies for DGARBN are potentially possible.

In this paper, two central issues of gene regulatory networks (see, e.g., [22]), attractor detection and optimal control, are well studied. We first formulate the extended state transition graph (ESTG) of a DGARBN. The state transition is encoded as a satisfiability modulo theory (SMT) formula. This formulation is a starting point for further analysis of DGARBNs in this paper. Next, we formally state and prove several relations between DGARBNs and other popular models including deterministic asynchronous models [15], block-sequential Boolean networks [4], generalized asynchronous random Boolean networks [5], and mixed-context random Boolean networks [11]. We then propose two SMT-based methods for attractor detection and optimal control of DGARBNs. These methods are implemented in a JAVA tool called DABoolNet. In order to highlight the scalability of the proposed methods, two experiments on randomly generated networks and one artificial network are conducted. To our best knowledge, DABoolNet is the first analytical and practical tool for attractor detection and optimal control of DGARBNs. In addition, several case studies are presented to show the applications of our methods. For attractor detection in DGARBNs, we apply DABoolNet to two real biological networks and compare the obtained results to the existing results in the literature. We also use DABoolNet to verify several insights into the dynamics of random Boolean networks presented in [5], [11]. For optimal control of DGARBNs, we apply DABoolNet to one real biological network.

The rest of this paper is organized as follows: Section 2 gives preliminaries on DGARBNs. Section 3 presents the formulation of ESTG. Section 4 presents the relations between DGARBNs and other models. Based on ESTG, the SMT-based method for finding attractors of DGARBNs is presented in Section 5. The proofs of all the lemmas and theorems in Sections 4 and 5 are presented in Supplemental material 1. Section 6 presents the formal definition and the SMT-based method for optimal control of DGARBNs. Results of the two experiments are shown in Section 7. Section 8 concludes the paper and discusses open problems.

2 Preliminaries

A deterministic generalized asynchronous random Boolean network (DGARBN) has a set of n nodes ($X = \{x_1, ..., x_n\}$). Each node x_i is associated with a Boolean function f_i ($f_i : \{0,1\}^n \rightarrow \{0,1\}$). Each node x_i is also associated with two parameters: $p_i \in \mathbb{N}^+$ and $q_i \in \mathbb{N}$ ($q_i < p_i$). p_i defines the period between two consecutive updates of node x_i while q_i determines the time to the first update of node x_i. We use $maxP$ as a parameter indicating the maximum allowed period of a node (i.e., $p_i \leq maxP, \forall i \in \{1, ..., n\}$). The set of all $p's$ and $q's$ is called the *context* of a DGARBN. Example 1 is an example of DGARBNs where "$\wedge$", "$\vee$", and "$\neg$" denote the Boolean logical operations CONJUNCTION, DISJUNCTION, and NEGATION, respectively.

Example 1. *A DGARBN includes two nodes ($X = \{x_1, x_2\}$). Its Boolean functions and context are given by:*

$$\begin{aligned} &f_1 = (x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_2), f_2 = x_1; \\ &p_1 = 1, p_2 = 2, q_1 = 0, q_2 = 0. \end{aligned} \tag{1}$$

$x_i(t) \in \{0,1\}$ denotes the value of node x_i at time t. A state of a DGARBN at time t is a vector $x(t) = (x_1(t), ..., x_n(t))$. At time t, node x_i will be updated by $x_i(t+1) = f_i(x(t))$ when the modulus of time t over p_i is equal to q_i (i.e., $t\%p_i = q_i$). For example, at time $t = 1$, only node x_1 of DGARBN (1) will be updated. If two or more nodes will be updated, they will be updated simultaneously (e.g., at time $t = 0$, node x_1 and node x_2 of DGARBN (1) will be updated simultaneously). Then, the current state $x(t)$ will transit to the next state $x(t+1)$. This is a state transition. The dynamics of a DGARBN is deterministic since $x(t+1)$ is uniquely determined for given $x(t)$. Especially, if all $p's$ are 1, then the DGARBN becomes a CRBN in which all the nodes will be updated simultaneously at any time t [5].

The dynamics of a DGARBN depends on the initial state (the state of the DGARBN at time $t = 0$). There are 2^n possible initial states. Since the evolution of DGARBNs is based on modulus arithmetic, we only need to consider the scaled time t_{scaled} of t to LCM (i.e., $t_{scaled} = t\%LCM$) where LCM is the least common multiple of all $p's$ [5]. Indeed, the pattern of updating nodes is repeated after each LCM time steps. Note that the visited states are not necessarily repeated. Figure 1 shows the dynamics of DGARBN (1). Herein, circles denote states while dashed circles denote initial states of the DGARBN. An arc and its above text denote a state transition and the scaled time of t when the state transition occurs, respectively. The DGARBN is updated in a pattern with period of 2: x_1 and x_2 together, x_1 alone. If DGARBN (1) starts with state 10, then x_1 and x_2 will be updated simultaneously leading to state 01. Next, x_1 will be updated leading to state 01. In the next time step, the pattern is repeated (i.e., x_1 and x_2 will be updated simultaneously). However, the next state (i.e., 00) has not been visited.

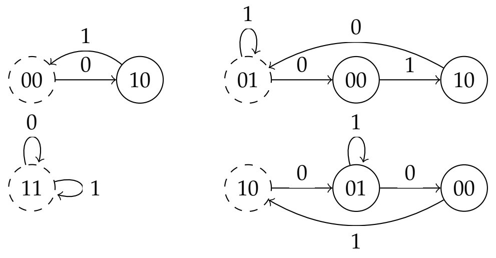

Fig. 1: Dynamics of DGARBN (1).

Start from an initial state, the DGARBN will eventually lead to an *attractor* since the number of states of the DGARBN is finite. Definition 1 is a general definition of an attractor for all types of BNs. An attractor is said to be a fixed point or a cyclic attractor if it consists of only one state or at least two states, respectively. The fixed points of a BN are the same regardless of its updating scheme. The cyclic attractors of deterministic (e.g., DGARBNs) and non-deterministic (e.g., ARBNs) asynchronous

BNs can be different. In non-deterministic asynchronous BNs, the system oscillates irregularly among the states of a cyclic attractor due to the randomness involved in the updating scheme. In this case, the cyclic attractor is referred to as a loose attractor [8]. In deterministic asynchronous BNs, the system oscillates regularly among the states in a cyclic attractor due to the deterministic dynamics. In this case, the cyclic attractor is referred to as a limit cycle [23]. However, there is no systematic definition for limit cycles of a DGARBN. Thus, we still use the term of cyclic attractors for DGARBNs. Let reconsider DGARBN (1). If it starts from 11, it will reach a fixed point ($\{11\}$). If it starts from 00, it will reach a cyclic attractor ($\{00, 10\}$). If it starts from 01 or 10, it will reach the same cyclic attractor ($\{01, 00, 10\}$).

Definition 1. *An attractor of a BN is a set of states satisfying any state in this set can reach any state (including this state) in this set and cannot reach any state that is not in this set.*

3 Extended state transition graph

Although the definition of a DGARBN enables us to study the behavior of a GRN in the long run, it does not provide a systematic mean for its analysis. We here propose a synchronization method for DGARBNs, which provides a synchronous representation for the dynamics of a DGARBN. This method can pave potential ways for analysis and control of DGARBNs.

We define an extended state of a DGARBN, which includes a state of this DGARBN and the scaled time t_{scaled} of time t when reaching this state. We use $x_{n+1} \in \{0, ..., LCM-1\}$ to represent the value of t_{scaled}. We define the *image* of an extended state $es \in \{0,1\}^n \times \{0, ..., LCM-1\}$ as a state $[[es]]^I \in \{0,1\}^n$ satisfying $[[es]]^I_i = es_i, i = \{1, ..., n\}$. Furthermore, we have $[[ES]]^I = \bigcup_{es \in ES} [[es]]^I$ where ES is a set of extended states. For clarification, we denote an extended stated es by $([[es]]^I, es_{n+1})$. For example, $(00, 1)$ means that $x_1 = 0$, $x_2 = 0$, and $x_3 = t_{scaled} = 1$. Then, the state transition formula is given as in (2) where x^j denotes the current extended state; x^{j+1} denotes the next extended state; "%" is the modulus operator; "=" is the logical predicate EQUALITY. Herein, the scaled time of the current extended state will increase by one; if it equals to LCM, then it is set to 0. This characteristic is presented by $x^{j+1}_{n+1} = (x^j_{n+1} + 1)\% LCM$. The value of the i-th node will be updated if the modulus of time t over p_i is equal to q_i and be not changed otherwise. The former case is presented by $(x^j_{n+1} \% p_i = q_i \wedge (x^{j+1}_i = f_i(x^j))$ while the latter case is presented by $(x^j_{n+1} \% p_i \neq q_i \wedge (x^{j+1}_i = x^j_i))$. Moreover, this state transition formula is in form of an SMT formula in infix notation [24]. Note that each x^j_i $(i = 1, ..., n)$ corresponds to a Boolean SMT variable and each x^j_{n+1} corresponds to an integer SMT variable.

$$\begin{aligned} T(x^j, x^{j+1}) \equiv \{ & x^{j+1}_{n+1} = (x^j_{n+1} + 1)\% LCM\} \wedge \\ & \bigwedge_{i=1}^{n} \{(x^j_{n+1} \% p_i = q_i \wedge (x^{j+1}_i = f_i(x^j)) \vee \\ & (x^j_{n+1} \% p_i \neq q_i \wedge (x^{j+1}_i = x^j_i))\} \end{aligned} \tag{2}$$

Then, the dynamics of a DGARBN is captured by an extended state transition graph (ESTG). An ESTG is a directed graph whose nodes and arcs denote extended states and state transitions, respectively. A state transition from extended state es to extended state es' of this ESTG is characterized by (2) (i.e., $T(es, es') = true$). Hereafter, we discuss about the number of extended states of this ESTG. Since $x_{n+1} \in \{0, ..., LCM-1\}$, we can have $LCM \times 2^n$ possible extended states. However, the number of possible initial extended states is only 2^n instead of $LCM \times 2^n$ since in an initial extended state, x_{n+1} is 0, i.e., the start time is always 0. Thus, the ESTG may have less than $LCM \times 2^n$ extended states. The extended states, which are not in the ESTG, are called *spurious* extended states. For example, let consider a DGARBN of two nodes ($X = \{x_1, x_2\}$). Its Boolean functions and context are given by (3). Its ESTG is given in Figure 2(right). As we can see, the extended state $(00, 1)$ is not in this ESTG, thus it is a spurious extended state.

$$\begin{aligned} & f_1 = x_1 \vee (\neg x_1 \wedge x_2), f_2 = (x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_2) \\ & p_1 = 2, p_2 = 1, q_1 = 0, q_2 = 0 \end{aligned} \tag{3}$$

By the definition of ESTG, each extended state in the ESTG of a DGARBN has exactly one successor extended state. In the case $LCM = 1$, two consecutive extended states of the ESTG may be the same since t_{scaled} is always 0. Thus, the ESTG can have fixed points or limit cycles. We define a limit cycle of length $p > 1$ as the sequence $es^0, ..., es^{p-1}$ of extended states such that es^j are pairwise distinct, es^{j+1} is the next extended state of es^j in the ESTG for all $j \in \{0, p-2\}$, and es^0 is the next extended state of es^{p-1} in the ESTG. Note that a fixed point can be seen as a limit cycle of length 1. In the case $LCM > 1$, two consecutive extended states are always different since they at least differ in t_{scaled}. Thus, the ESTG has only limit cycles. Moreover, based on (2), we can easily imply that the length of a limit cycle of the ESTG is a multiple of LCM.

Since an ESTG can capture the whole dynamics of a DGARBN, the attractors of the DGARBN are represented by fixed points or limit cycles of its ESTG. Since a fixed point may only appear when $LCM = 1$, the length of an attractor (i.e., the length of a fixed point or a limit cycle) is a multiple of LCM. Especially, a fixed point $\{s\}$ of the DGARBN is represented by a limit cycle c of its ESTG where the length of c is LCM and any extended state es in c satisfies $[[es]]^I = s$.

Figure 2(left) shows the ESTG of DGARBN (1). Since $LCM = 2$, this ESTG has no fixed points. This ESTG has three limit cycles including $\{(11, 0), (11, 1)\}$, $\{(00, 0), (10, 1)\}$, and $\{(01, 0), (00, 1), (10, 0), (01, 1)\}$. $\{(11, 0), (11, 1)\}$ corresponds to the fixed point $\{11\}$ of DGARBN (1) while $\{(00, 0), (10, 1)\}$ and $\{(01, 0), (00, 1), (10, 0), (01, 1)\}$ correspond to the two cyclic attractors $\{00, 10\}$ and $\{01, 00, 10\}$ of DGARBN (1), respectively.

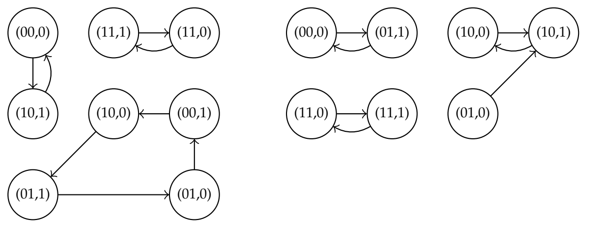

Fig. 2: (left) ESTG of DGARBN (1). (right) ESTG of DGARBN (3).

4 Relations with other models

In this section, we will analyze relations between DGARBNs and other models, such as, deterministic asynchronous (DA) models [15], [25], [13], block-sequential Boolean networks [26], [17], generalized asynchronous random Boolean networks [5], [27], and mixed-context random Boolean networks [11]. The obtained relations are theoretical findings contributing to understanding the dynamics of random Boolean networks. These findings also pave the potential ways to analyze these other models based on DGARBNs.

4.1 DGARBNs and DA models

In a DA model, each node x_i is associated with a pre-selected time unit $\gamma_i \geq 1$ [13]. The update of a node depends on the current time step by (4). If multiple nodes can update at time t, then they will update synchronously.

$$x_i(t+1) = \begin{cases} f_i(x(t)) & \text{if } t+1 = k\gamma_i, k \in \{1,2,...\} \\ x_i(t) & \text{otherwise} \end{cases} \quad (4)$$

Definition 2. *Given a DA model as in (4). Let $\mathcal{D}$ be its corresponding DGARBN. $\mathcal{D}$ and the DA model share the same sets of nodes and Boolean functions. The context of $\mathcal{D}$ is as follows: $p_i = \gamma_i, q_i = p_i - 1, i \in \{1, ..., n\}$.*

Obviously, a DA model and its corresponding DGARBN by Definition 2 have the same behaviors. Indeed, assume that the node x_i of the DA model will be updated at time t, i.e., $t+1 = k\gamma_i, k \in \{1,2,...\}$. In the DGARBN, $t\%p_i = (k\gamma_i - 1)\%\gamma_i = \gamma_i - 1 = q_i$, thus x_i will also be updated at time t. This finding allows us to directly apply the methods for DGARBNs to DA models.

4.2 DGARBNs and block-sequential Boolean networks

A block-sequential Boolean network (BSBN) [26], [17] is a tuple $\langle V, F, d\rangle$ where $V = \{x_1, ..., x_n\}$ is the set of nodes, $F = \{f_1, ..., f_n\}$ is the set of Boolean functions associated with the nodes, and d is a deterministic updating scheme (see Definition 3). At each time step, nodes in a block are updated in parallel, but blocks follow each other sequentially along with d, leading to a new state. Since each state has only one outgoing transition, a BSBN may have two types of attractors including fixed points and limit cycles [26]. Note that the time unit of a BSBN is $nb(d)$ times of the time unit of CRBNs or DGARBNs. Example 2 shows an example of BSBNs.

Definition 3 (Adapted from [26]). *A (deterministic) updating scheme on the set V of n nodes is a function $d : \{1, ..., n\} \to \{1, ..., m\}, m \leq n$. A* block *of d is the set $B_i = \{v \in V | d(v) = i\}, 1 \leq i \leq m$. The number of blocks of d is denoted by $nb(d) \equiv m$. Frequently, d will be denoted as a sequence of blocks, i.e., $d = (j \in B_1)(j \in B_2)...(j \in B_{nb(d)})$.*

Example 2. *Let $\mathcal{B} = \langle V, F, d\rangle$ be a BSBN where V and F are given as in DGARBN (1) and $d = (x_1)(x_2)$. The STG of $\mathcal{B}$ is given in Figure 3(left). For example, in state 00, x_1 is updated, leading to a new state 10, then x_2 is updated, leading to a new state 11. Now, $(00, 11)$ is a state transition of $\mathcal{B}$. As we can see, $\mathcal{B}$ has only one fixed point ($\{11\}$).*

Definition 4. *Let $\mathcal{B} = \langle V, F, d\rangle$ be a BSBN. Then, $\mathcal{D}$ is a DGARBN such that its set of nodes is V and its set of Boolean functions is F and its context is given as: $p_i = nb(d), q_i = j - 1, x_i \in B_j, i \in \{1, ..., n\}$. Given a state s of $\mathcal{B}$, the corresponding extended state of s is $[[s]]^{\mathcal{D}}$ where $[[s]]^{\mathcal{D}}_i = s_i, [[s]]^{\mathcal{D}}_{n+1} = 0, i \in \{1, ..., n\}$.*

Definition 4 gives the encoding of a BSBN as a DGARBN. Figure 3(right) shows the ESTG of the encoded DGARBN of the BSBN in Example 2. This ESTG has one limit cycle of length 2 ($\{(11, 0), (11, 1)\}$). As we can see, a state s reaches a state s' in

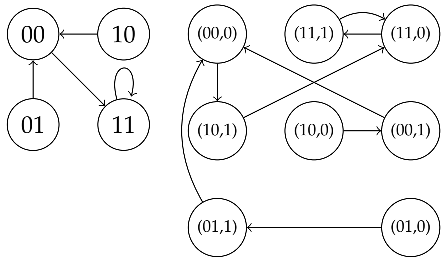

Fig. 3: (left) STG of the BSBN. (right) ESTG of the encoded DGARBN.

the BSBN if and only if the corresponding extended state of s reaches the corresponding extended state of s' in the encoded DGARBN. For example, 00 reaches 11 if and only if $(00, 0)$ reaches $(11, 0)$. We can also see that the set of attractors of the BSBN one-to-one corresponds to the set of attractors of the encoded DGARBN. Hereafter, we formalize the relations between a BSBN and its encoded DGARBN. These relations allow us to apply our methods for DGARBNs to BSBNs.

Theorem 1. *Let $\mathcal{B} = \langle V, F, d\rangle$ be a BSBN and $\mathcal{D}$ be its encoded DGARBN by Definition 4. For any pair of states s and s', we have s' is reachable from s in $\mathcal{B}$ iff $[[s']]^{\mathcal{D}}$ is reachable from $[[s]]^{\mathcal{D}}$ in $\mathcal{D}$.*

Theorem 2. *Let $\mathcal{B} = \langle V, F, d\rangle$ be a BSBN and $\mathcal{D}$ be its encoded DGARBN by Definition 4. Let $A^{\mathcal{B}}$ and $A^{\mathcal{D}}$ be the sets of attractors of $\mathcal{B}$ and $\mathcal{D}$, respectively. Then, $A^{\mathcal{B}}$ one-to-one corresponds to $A^{\mathcal{D}}$. Moreover, given an attractor att of $\mathcal{B}$, its corresponding attractor of $\mathcal{D}$ is att' satisfying $att = [[att']]^{\mathcal{B}}$ where $[[ES]]^{\mathcal{B}} = \{s | (s, 0) \in ES\}$, ES is a set of extended states.*

4.3 DGARBNs and GARBNs

Generalized Asynchronous Random Boolean Networks (GARBNs) are interesting mathematical objects and have been widely studied [5], [21], [27], [28]. GARBNs have the asynchronous and non-deterministic updating scheme. At each time step, they randomly select any number of nodes to update synchronously. This means that a GARBN can update synchronously no node, only one node, some nodes, or all the nodes. The STG of a GARBN has 2^n nodes and up to 2^{2n} arcs [5]. Thus, GARBNs are more complex than CRBNs. This maybe makes the analysis of a GARBN more computationally complex than that of its CRBN counterpart [27].

Example 3. *Let $\mathcal{G}$ be the GARBN counterpart of DGARBN (1) (i.e., the GARBN shares the sets of nodes and Boolean functions with DGARBN (1)). The STG of $\mathcal{G}$ is shown in Figure 4. As we can see, each arc of the ESTG of DGARBN (1) (see Figure 2(left)) corresponds to an arc of the STG of $\mathcal{G}$. For example, $((00, 0), (10, 1))$ corresponds to $(00, 10)$ and $((10, 0), (01, 1))$ corresponds to $(10, 01)$. $\mathcal{G}$ has only one attractor ($\{11\}$) while DGARBN (1) has three attractors. We can see that $\{11\}$ corresponds to the attractor $\{(11, 1), (11, 0)\}$ of DGARBN (1).*

Obviously, all state transitions of a DGARBN will be covered in the STG of its GARBN counterpart by the updating scheme of GARBNs. See Example 3 as an illustration. Hereafter, we formally present several relations between a DGARBN and its GARBN counterpart in Lemma 1 and Theorem 3. Theorem 3 shows that given a DGARBN and its GARBN counterpart, each attractor of the GARBN contains at least one attractor of the DGARBN.

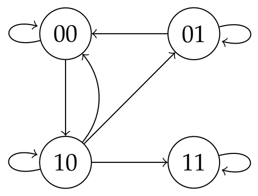

Fig. 4: STG of the GARBN counterpart of DGARBN (1).

Lemma 1. *Let $\mathcal{D}$ be a DGARBN and $\mathcal{G}$ be its GARBN counterpart. Let es be an extended state of $\mathcal{D}$ and $FR^{\mathcal{D}}(\{es\})$ be the set of extended states reachable from es in $\mathcal{D}$. Then, $[[FR^{\mathcal{D}}(\{es\})]]^I \subseteq FR^{\mathcal{G}}(\{[[es]]^I\})$.*

Theorem 3. *Let $\mathcal{D}$ be a DGARBN and $\mathcal{G}$ be its GARBN counterpart. Let $A^{\mathcal{D}}$ and $A^{\mathcal{G}}$ be the sets of attractors of $\mathcal{D}$ and $\mathcal{G}$, respectively. Then, there exists a mapping $m : A^{\mathcal{G}} \to A^{\mathcal{D}}$ with $[[m(att)]]^I \subseteq att$ for all $att \in A^{\mathcal{G}}$. Moreover, $m(att_1) \neq m(att_2)$ for all $att_1, att_2 \in A^{\mathcal{G}}, att_1 \neq att_2$. That means m is an injection.*

In [27], we have stated and formally proved several relations between attractors of a CRBN and attractors of its GARBN counterpart. Theorem 5 of [27] shows that any attractor of a GARBN always contains an attractor of its CRBN counterpart. Since a CRBN is a special DGARBN, Theorem 5 of [27] is a special case of Theorem 3.

4.4 DGARBNs and MxRBNs

To deal with the lack of knowledge on real contexts of DGARBNs, Carlos Gershenson proposed a new type of Boolean models called mixed-context random Boolean networks (MxRBNs) [11], [2]. He introduced the idea of *mixed context*, where a mixed context is a statistical mixture of a set of pure contexts. The updating scheme of MxRBNs is basically like that of DGARBNs. However, MxRBNs have to make a random choice between pure contexts at each time step, making their dynamics non-deterministic and generating a probability structure that is non-Kolmogorovian (quantum-like) [11].

An MxRBN is a DGARBN with $M \geq 1$ *pure contexts*. Each pure context consists of p's and q's as in DGARBNs. At each time step, we randomly select one pure context among M pure contexts and use it in the DGARBN. Hence, the dynamics of an MxRBN can be captured by overlapping the ESTGs of its M constituent DGARBNs. See Example 4 for an example of MxRBNs.

Example 4. *Given an MxRBN $\mathcal{M}$ of two nodes ($X = \{x_1, x_2\}$). Its Boolean functions and $M = 2$ pure contexts are given by:*

$$\begin{aligned} &f_1 = (x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_2), f_2 = x_1;\\ &M_1 : p_1 = 1, p_2 = 2, q_1 = 0, q_2 = 0; \\ &M_2 : p_1 = 1, p_2 = 1, q_1 = 0, q_2 = 0. \end{aligned} \tag{5}$$

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be the DGARBNs corresponding to M_1 and M_2, respectively. Figure 5 shows the ESTG of $\mathcal{M}$. This ESTG is formed by overlapping the ESTG of $\mathcal{D}_1$ (normal lines) and the ESTG of $\mathcal{D}_2$ (dashed lines).

In Example 4, $\mathcal{M}$ has two attractors including {(11,0), (11,1)} and $\{(00,0), (10,1), (10,0), (01,1), (01,0), (00,1)\}$. Note that an extended state of an MxRBN may have more than one outgoing transition by the introduction of non-determinism to MxRBNs, leading to the existence of complex attractors as in ARBNs [7]. Moreover, $\mathcal{D}_1$ has three attractors including $\{(00,0), (10,1)\}$, $\{(11,0), (11,1)\}$, and $\{(10,0), (01,1), (01,0), (00,1)\}$; $\mathcal{D}_2$ has two attractors including $\{(11,0)\}$ and {(00,0), (10,0), (01,0)}. We can see that each attractor of $\mathcal{M}$ always contains at least one attractor of $\mathcal{D}_1$. This observation is also valid for the case of $\mathcal{D}_2$. We generalize this observation in Theorem 4.

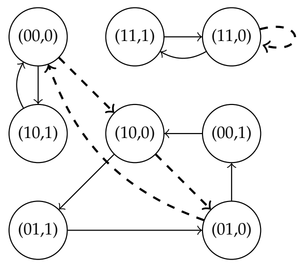

Fig. 5: ESTG of MxRBN (5).

Theorem 4. *Let $\mathcal{M}$ be an MxRBN of M pure contexts. Let $\mathcal{D}$ be an arbitrary DGARBN among M constituent DGARBNs of $\mathcal{M}$. Let $\mathcal{A}^{\mathcal{M}}$ and $\mathcal{A}^{\mathcal{D}}$ be the sets of attractors of $\mathcal{M}$ and $\mathcal{D}$, respectively. Then, there exists a mapping $m : \mathcal{A}^{\mathcal{M}} \to \mathcal{A}^{\mathcal{D}}$ with $m(att) \subseteq att$ for all $att \in \mathcal{A}^{\mathcal{M}}$. Moreover, $m(att_1) \neq m(att_2)$ for all $att_1, att_2 \in \mathcal{A}^{\mathcal{M}}, att_1 \neq att_2$. That means m is an injection.*

Theorem 4 suggests us a promising way to find all attractors of an MxRBN. First, we can find attractors of one of its constituent DGARBNs by applying our method presented in Section 5. Then, we can filter the set of DGARBN attractors by checking the reachability property in this MxRBN. This idea is similar to the filtering algorithm for finding GARBN attractors based on CRBN attractors, which is presented in [27]. Proposing an efficient method for attractor detection in MxRBNs is one of our future work.

5 Attractor Detection

Attractors correspond to steady states which are important long-term behaviors of GRNs. Attractors of GRNs are linked to phenotypes [1] and functional cellular states, such as, proliferation, apoptosis, or differentiation [29]. Therefore, in addition to the usage in understanding GRNs, analysis of attractors could provide new insights into systems biology (e.g., the origins of cancer) [30]. Attractors also play a crucial role in the development of new drugs [31].

Attractor detection in various types of BNs has attracted much attention. Many studies has been done but they mainly focus on CRBNs (e.g., [7], [32], [33]) and ARBNs (e.g., [7], [34], [35]). There are very few studies (e.g., [18], [25]) specifically done for DGARBNs. However, they are theoretical or simulation-based studies. This motivates algorithms for analytically and practically finding attractors of DGARBNs.

5.1 Satisfiability modulo theory-based method

Dubrova and Teslenko proposed an efficient SAT-based method for finding attractors of CRBNs [32]. Since the ESTG of a DGARBN is deterministic like the state transition graph (STG) of a CRBN, it is potential to extend the SAT-based method to that for DGARBNs. However, it is difficult to directly use SAT for DGARBNs because the state transition formula (2) contains integer variables (e.g., x^j_{n+1}) and the modulus operator. Since integers are bounded, we can encode x^j_{n+1} by a bit-vector [24] of size m (m is the smallest integer larger than or equal to the logarithm to the base 2 of LCM) and then use the bit-blasting technique [36] to transform (2) to an equivalent SAT formula. Satisfiability Modulo Theory (SMT) may provide a more natural encoding. Modern SMT solvers (e.g., Z3 [24]) use the efficient algorithms of SAT solvers as their solving cores, thus can handle very large problem instances. Moreover, the bit-blasting technique is also integrated into some SMT solvers as a solving tactic [24]. Therefore, we here propose a new SMT-based method for finding attractors of DGARBNs. Note that proposing ESTG paves the way to extend the method by Dubrova and Teslenko for CRBNs to that for DGARBNs.

The intuitive idea of the proposed SMT-based method is as follows. We search for a p-length path (i.e., this path makes p transitions between extended states) in the ESTG of a DGARBN. Since a fixed point of the DGARBN is represented by a limit cycle of length LCM in the ESTG of this DGARBN, p can start with LCM ($LCM \geq 1$). If a path is found and it contains a limit cycle, we add this limit cycle to the set of marked attractors. In an ESTG, an extended state has a unique next extended state, thus once a path reaches a limit cycle, it never leaves this cycle. This suggests a way to check whether a path contains a limit cycle by checking whether the last extended state of the path occurs at least twice in this path. In the next iterations, we only search for paths such that their last extended states are not in the set of marked attractors. If a path is found and it is cycle-free, we increase p (e.g., double p) and continue the search for a path with the new length. If a path does not exist, we can terminate the search. Since the ESTG is deterministic like the STG of a CRBN, the proposed SMT-based method always terminates and correctly finds all attractors of the DGARBN (see Theorem 5).

Theorem 5. *The proposed SMT-based method terminates and correctly finds all attractors of a DGARBN.*

In each iteration of our method, the finding of a p-length path can be performed by using an SMT solver (we use Z3 [24] in this paper). Let A be the set of attractors which is updated through the iterations. Note that such a path must satisfy two conditions. Firstly, the value of the scaled time of its start extended state must be 0. As mentioned in Section 3, there may be some spurious extended states which are not in the ESTG of the DGARBN. This condition guarantees that all extended states along with the path are always in the ESTG since the start extended state is one of the possible initial extended states of the DGARBN. Secondly, its last extended state must be not in A as mentioned in the previous paragraph. The path can be encoded as an SMT formula P (6). Then, we simply use Z3 to solve P.

$$P \equiv (x^0_{n+1} = 0) \wedge \bigwedge_{j=0}^{p-1} T(x^j, x^{j+1}) \wedge \neg\chi(A^F, x^p) \tag{6}$$

In (6), x^j denotes the $(j+1)$-th extended states of the path. Then, x^0 and x^p denote the start and end extended states of the p-length path, respectively. $T(x^j, x^{j+1})$ represents the state transition from x^j to x^{j+1} where T is the transition formula of the DGARBN (see (2)). Thus, $\bigwedge_{j=0}^{p-1} T(x^j, x^{j+1})$ represents a p-length path of the DGARBN. $(x^0_{n+1} = 0)$ represents the first condition of the path. $\neg\chi(A^F, x^p)$ represents the second condition of the path where A^F is the flattened set of A (i.e., A^F is the set of extended states); $\chi(A^F, x^p)$ is the characteristic formula representing all extended states of A^F in terms of variables of x^p. The characteristic formula of a set of extended states is defined based on the characteristic formula of an extended state: $\chi(A^F, x^p) = \bigvee_{s \in A^F} \chi(s, x^p)$. The characteristic formula of an extended state s in terms of variables of x^p is defined as $\chi(s, x^p) = \bigwedge_{i=1}^{n+1}(s_i = x^p_i)$.

Let see a running example on DGARBN (1). Our method starts with $p = LCM = 2$ and $A = \emptyset$. We have:

$$\begin{aligned}
&T(x^0, x^1) = \{x^1_3 = (x^0_3 + 1)\%2\} \wedge \{(x^0_3\%1 = 0 \wedge \\
&(x^1_1 = (x^0_1 \wedge x^0_2) \vee (\neg x^0_1 \wedge \neg x^0_2))) \vee (x^0_3\%1 \neq 0 \wedge x^1_1 = x^0_1)\} \\
&\wedge \{(x^0_3\%2 = 0 \wedge x^1_2 = x^0_1) \vee (x^0_3\%2 \neq 0 \wedge x^1_2 = x^0_2)\}; \\
&T(x^1, x^2) = \{x^2_3 = (x^1_3 + 1)\%2\} \wedge \{(x^1_3\%1 = 0 \wedge \\
&(x^2_1 = (x^1_1 \wedge x^1_2) \vee (\neg x^1_1 \wedge \neg x^1_2))) \vee (x^1_3\%1 \neq 0 \wedge x^2_1 = x^1_1)\} \\
&\wedge \{(x^1_3\%2 = 0 \wedge x^2_2 = x^1_1) \vee (x^1_3\%2 \neq 0 \wedge x^2_2 = x^1_2)\}; \\
&P = (x^0_3 = 0) \wedge T(x^0, x^1) \wedge T(x^1, x^2).
\end{aligned}$$

We then use Z3 to solve P. Clearly, a path is found. Suppose that this path is $(00, 0) \to (10, 1) \to (00, 0)$ (see Figure 2(left)). Since the last extended state ($(00, 0)$) occurs twice, we obtain a limit cycle of length 2. Now, we can add this limit cycle ($\{(00, 0), (10, 1)\}$) to A. The flatted set A^F is $\{(00, 0), (10, 1)\}$ and $\chi(A^F, x^2) = \chi((00, 0), x^2) \vee \chi((10, 1), x^2)$ where $\chi((00, 0), x^2) = (x^2_1 = 0) \wedge (x^2_2 = 0) \wedge (x^2_3 = 0)$, $\chi((10, 1), x^2) = (x^2_1 = 1) \wedge (x^2_2 = 0) \wedge (x^2_3 = 1)$. In the next iteration, our method continues to search a path of length 2 with the new formula $P = (x^0_3 = 0) \wedge T(x^0, x^1) \wedge T(x^1, x^2) \wedge \neg\chi(A^F, x^2)$. Suppose that the next found path is $(00, 1) \to (10, 0) \to (01, 1)$. This path is cycle-free since the last extended state ($(01, 1)$) occurs only one. Now, we increase p to 4 and start the next iteration with $p = 4$ and $A = \{\{(00, 0), (10, 1)\}\}$. By running two more iterations, our method terminates and all attractors are detected with A = {{(00,0), (10,1)}, {(11,0), (11,1)}, {(00,1), (10,0), (01,1), (01,0)}}.

In the proposed SMT-based method, the last value of p is called the unfolding depth of the DGARBN. Obviously, in each iteration of the method, the number variables and the number of clauses of P depend on both n and p. In the ESTG of the DGARBN, starting from an extended state, we must go through at least $LCM - 1$ different extended states to reach an extended state with the same t_{scaled}. It seems to make the diameter of the ESTG longer. Thus, if LCM is large even when n is small, the diameter may be very large. The diameter of the ESTG is an upper bound of the unfolding depth, leading the unfolding depth of the DGARBN may be very large. In this case, P will have too many variables and clauses, and the time for solving P may be extremely long.

5.2 Case study

In this subsection, we apply our method for attractor detection in DGARBNs to two real biological networks and compare the obtained results to the existing results on these networks in the literature.

The first network is the reduced network of the guard cell ABA signal transduction network analyzed in [13]. The BN of this reduced network includes three nodes including x_1 standing for CIS, x_2 standing for Ca^{2+}_c, and x_3 standing for Ca^{2+}ATPase. Its Boolean functions are given by: $f_1 = x_2, f_2 = x_1 \wedge \neg x_3, f_3 = x_2$. Each node x_i is associated with a time unit γ_i, forming a DA model. Saadatpour et al. applied their method to this network with many different choices of time units (i.e., many different DA models). We here encode a DA model as a DGARBN (see Subsection 4.1). Then, we apply our SMT-based

TABLE 1: Boolean functions of the cell cycle network.

Gene	Boolean function
$CycD$	$CycD$
Rb	$(\neg CycD \wedge \neg CycB) \wedge ((\neg CycE \wedge \neg CycA) \vee p27)$
$E2F$	$(\neg Rb \wedge \neg CycA \wedge \neg CycB) \vee (p27 \wedge \neg Rb \wedge \neg CycB)$
$CycE$	$(E2F \wedge \neg Rb)$
$CycA$	$(\neg Rb \wedge \neg Cdc20 \wedge \neg(Cdh1 \wedge UbcH10)) \wedge (E2F \vee CycA)$
$p27$	$(\neg CycD \wedge \neg CycB) \wedge ((\neg CycE \wedge \neg CycA) \vee (p27 \wedge \neg(CycE \wedge CycA)))$
$Cdc20$	$CycB$
$Cdh1$	$(\neg CycA \wedge \neg CycB) \vee Cdc20 \vee (p27 \wedge \neg CycB)$
$UbcH10$	$\neg Cdh1 \vee (Cdh1 \wedge UbcH10 \wedge (Cdc20 \vee CycA \vee CycB))$
$CycB$	$\neg Cdc20 \wedge \neg Cdh1$

method to find attractors of the encoded DGARBN. The results are given as follows.

For the case $\gamma_1 = 2\gamma_2, \gamma_2 = 2k+1, \gamma_3 = 2, k = 2$, we have the DA model has a limit cycle of length $\gamma_2 + 1 + 2k = 4k+2 = 10$ by Proposition 1 of [13]. Note that in [13], the authors claim the cycle length is only $2k+2$ because of their way for counting the time staying each state. Following their proof for this proposition, the correct length should be $4k+2$. By applying our method, the encoded DGARBN has two limit cycles of length 10. One of these two limit cycles is {(101,0), (111,9), (111,8), (111,7), (111,6), (110,5), (100,4), (100,3), (100,2), (101,1)}. This result is consistent with Proposition 1 of [13].

For the case $\gamma_1 = 2\gamma_2, \gamma_2 = 2k, \gamma_3 = 2, k = 2$, we have the DA model has a limit cycle of length $\gamma_2 + 2k = 4k = 8$ by Proposition 2 of [13]. Note that in [13], the authors claim the cycle length is only $2k$ because of their way for counting the time staying each state. Following their proof for this proposition, the correct length should be $4k$. By applying our method, the encoded DGARBN has two limit cycles of length 8. One of these two limit cycles is {(110,5), (110,4), (100,3), (100,2), (101,1), (101,0), (111,7), (111,6)}. This result is consistent with Proposition 2 of [13].

The second network is the mammalian cell cycle network analyzed in [17]. The BN of this network includes 10 nodes (genes) and its Boolean functions are given in Table 1. The authors of [17] analyzed this network under different deterministic updating schemes. For each deterministic updating scheme, we encode the corresponding BSBN as a DGARBN (see Subsection 4.2). Then, we apply our SMT-based method to find attractors of the encoded DGARBN. The results are given as follows.

For the case the deterministic updating scheme is $(CycD, Rb, Cdc20, Cdh1, CycA)(p27, UbcH10, CycB)$ $(E2F)(CycE)$, the BSBN has one fixed point with $CycD = 0$, a limit cycle of length 4 with $CycD = 1$, and a limit cycle of length 8 with $CycD = 0$ (see Figure 7 of [17]). By applying our method, the encoded DGARBN has a limit cycle of length 4, a limit cycle of length 16, and a limit cycle of length 32. Since LCM of the encoded DGARBN is 4, three attractors of the encoded DGARBN correspond to three attractors of the BSBN. See Table 2 for details of these DGARBN attractors. Herein, the order of the nodes in a state is $(CycD, p27, E2F, CycE, CycA, p27, Cdc20, Cdh1, UbcH10,$ $CycB)$. Note that in each DGARBN attractor, we only show the images of extended states whose t_{scaled} are 0. The result is consistent with the relations presented in Subsection 4.2.

For the case the deterministic updating scheme is $(CycD, p27, Cdc20, Cdh1, UbcH10, CycB)(E2F)(CycE)$ $(Rb, CycA)$, the BSBN has one fixed point with $CycD = 0$, a limit cycle of length 2 with $CycD = 0$, a limit cycle of length 6 with $CycD = 0$, and a limit cycle of length 6 with $CycD = 1$ (see Figure 8 of [17]). By applying our method, the encoded DGARBN has a limit cycle of length 4, a limit cycle of length 8, and two limit cycles of length 24. Since LCM of the encoded DGARBN is 4, four attractors of the encoded DGARBN correspond to four attractors of the BSBN. See Table 2 for details of these DGARBN attractors. The result is consistent with the relations presented in Subsection 4.2.

TABLE 2: Details of DGARBN attractors.

Figure 7	{0100010100}
	{1011000100, 1000101010, 1000100011, 1000100100}
	{0000100100, 0011000100, 0011001010, 0000010011, 0100100100, 0011010100, 0000101010, 0000100011}
Figure 8	{0100010100}
	{0111110100, 0000000100}
	{0000100011, 0000100000, 0011100100, 0011000110, 0011001110, 0000001011}
	{1011001110, 1000001011, 1000100011, 1000100000, 1011100100, 1011000110}

5.3 Verifying the existing insights

Many numerical experiments were conducted to discover sevaral insights into the dynamics of DGARBNs [5], [11]. However, their method is incomplete and is limited to small networks ($n \leq 10$). Since DABoolNet can find exactly all attractors of a DGARBN and can be applied to large networks (see Subsection 7.1), we here reproduce these experiments with larger networks to verify these insights. We hope that our method will be helpful in further research on DGARBNs and their applications.

Following the experimental method by [5], [11], we randomly generated 1000 N-K BNs with $K = 3$ (i.e., each node has exactly $K = 3$ input nodes) and different numbers of nodes ($n = 3, 4, ..., 15$) by using Bool Net R package [37]. We then randomly generated a context for each BN with $maxP = 3$. In total, we have 13000 randomly generated DGARBNs. We applied the SMT-based method to find attractors of each DGARBN and its CRBN counterpart (i.e., $maxP = 1$). For each DGARBN or its CRBN counterpart, the number of attractors and the percentage of extended states in attractors were reported.

Figure 6 shows the average number of attractors of the randomly generated networks with different n. We here can see that the average number of attractors of DGARBNs for low $maxP$ ($maxP = 1$ and $maxP = 3$) has a linear increment proportional to n. This observation is consistent with the insight on the average number of attractors presented in Section 3 of [11].

Figure 7 shows the percentage of attractor states (in a base-10 logarithmic scale) of the randomly generated networks with different n. By observing this figure, we can see that the percentage of attractor states seems to decrease exponentially as n increases for both CRBNs and DGARBNs. However, the percentage of attractor states decreases slower for low $maxP$ ($maxP = 1$) than for a higher one ($maxP = 3$). This observation is consistent with the insight on the percentage of attractor states presented in Section 3 of [11]. Another observation, which can be obtained from this figure, is that CRBNs ($maxP = 1$) have more states in attractors than DGARBNs ($maxP = 3$). This is consistent with the insight on the percentage of attractor states presented in Subsection 3.2 of [5].

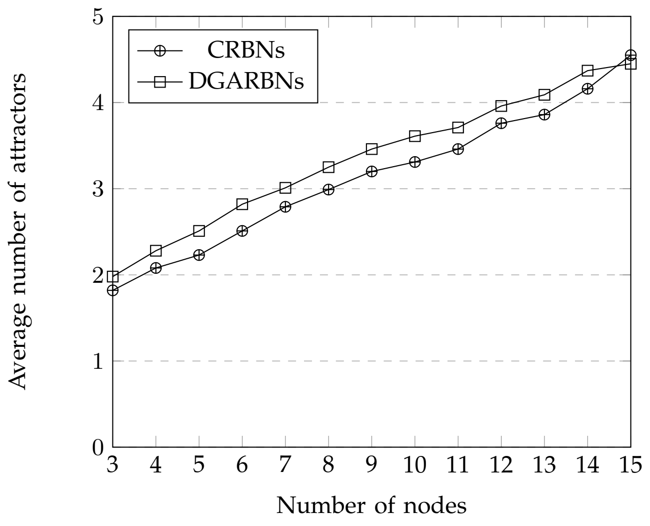

Fig. 6: Average number of attractors varying n.

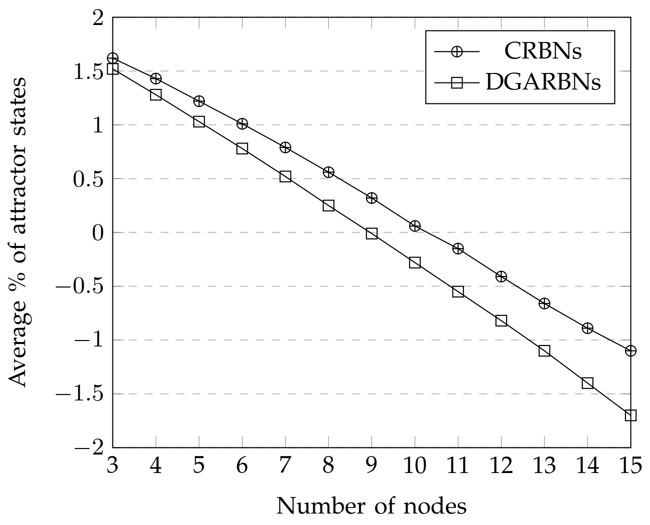

Fig. 7: Average percentage of attractor states n (log scale).

6 Optimal control

Control of biological systems is one of the central issues in systems biology [38]. In theory, biological systems are complex and contain highly non-linear components and thus existing methods in control theory cannot be directly applied to control of biological systems. In practice, control of cells may be useful for systems-based drug discovery and cancer treatment [38], [14], [39]. Thus, it is an important and interesting challenge to develop theories and methods for control of biological systems [22]. Since BNs are highly non-linear systems and has been widely used in modeling biological systems, it is reasonable to try to develop methods for control of BNs.

In recent years, several approaches have been developed for control of BNs. We can classify them into two main directions. The first one (e.g., [39], [40], [41], [42], [43], [44], [45]) uses node perturbations while the second one (e.g., [46], [47], [48]) uses external inputs to control a BN. The second direction assumes that the set of control inputs is known and fixed for a finite sequence of steps. Somehow this is not very realistic, concerning the effort in searching for potential drug targets [22]. However, this direction is still useful since (1) extensive studies have been done on selecting and analyzing the minimum set of control inputs [49], [50] and (2) some methods of the first direction can be cast into the second direction [41]. To our best knowledge, there is no study specially designed for DGARBNs. Although optimality may rarely be the main constraint compared to correct behaviors of systems in experiments, the optimal control problem is still useful and interesting since it is more general than the standard control problem [22], [48]. Therefore, we will focus on optimal control of DGARBNs.

We here formally define optimal control of DGARBNs as in Definition 5 adapted from [22]. In this definition, internal nodes stand for usual nodes (i.e., genes or proteins), control nodes can stand for external interventions (e.g., drugs, radiation, or chemotherapy), the initial state can stand for a disease or cancerous state, the target state can stand for a healthy or normal state. Usually, $u_i(k) = 0$ implies that u_i is not applied at time k, while $u_i(k) = 1$ implies that u_i is applied at time k with the application cost g_i. Note that the cost vector g may be fixed or not fixed over time. In the biological context, g may depend on the current state of the system, i.e., g may be represented as a function $g_f : U \times \{0,1\}^n \rightarrow \mathbb{N}$. Moreover, we can consider various types of cost, such as, the cost of applying drugs, the total harmful effects of the applied drugs, the total concentration of some given genes [51]. Representing the function g_f or these types of cost as SMT formulas is easy thanks to the expressive power of SMT. For simplicity, we assume that g is fixed over time and the cost function is the cost of applying control nodes. Example 5 shows a DGARBN with a setting for optimal control.

Definition 5. *Given a DGARBN including a set of internal nodes ($X = \{x_1, ..., x_n\}$) and a set of control nodes ($U = \{u_1, ..., u_m\}$), an initial state $s^{ini} \in \{0,1\}^{1\times n}$, a target state $s^{tar} \in \{0,1\}^{1\times n}$, a target time M, a cost vector $g \in \mathbb{N}^{1\times m}$. Note that control nodes can appear in Boolean functions of the DGARBN, i.e., $f_i : \{0,1\}^{n+m} \rightarrow \{0,1\}$ ($i = 1, ..., n$). Let decide whether or not there exists a control sequence of 0-1 control vectors $\langle u(0), ..., u(M-1)\rangle$ such that $x(0) = s^{ini}$, $x(M) = s^{tar}$, and the linear cost function $C = \sum_{j=0}^{M-1}(\sum_{i=1}^{m}(u_i(j) \times g(u_i)))$ is minimum. Then, output one if it exists.*

Example 5. *A DGARBN includes three internal nodes (x_1, x_2, x_3) and two control nodes (u_1, u_2). Its setting for optimal control is given by:*

$$\begin{aligned} &f_1 = \neg u_1, f_2 = x_1 \wedge u_2, f_3 = x_1 \vee x_2;\\ &p_1 = 2, p_2 = 1, p_3 = 2; q_1 = 1, q_2 = 0, q_3 = 0;\\ &g(u_1) = 1, g(u_2) = 2; s^{ini} = 000, s^{tar} = 011.\end{aligned} \tag{7}$$

We here consider two control modes (time-sensitive and non-time-sensitive) for optimal control of DGARBNs as for optimal control of other systems [52]. In the time-sensitive mode, the condition $x(M) = s^{tar}$ must be strictly satisfied, i.e., the DGARBN must reach the target state at exactly the target time M (as in Definition 5). The time-sensitive mode is often used in standard forms of optimal control problems of BNs [53]. It is useful when we want, for example, to find a drug treatment over a given time frame $[0, M]$ minimizing the cost of using the drugs. In many applications, we may want, for example, to find a drug treatment such that the cost of using the drugs is minimum and the treatment time should be as early as possible before a given time M. In this case, the non-time-sensitive mode is useful. In the non-time-sensitive mode, the condition $x(M) = s^{tar}$ can be relaxed, i.e., the DGARBN can reach the target state before or at time step M. Specifically, the two last sentences of Definition 5 are replaced by: Let decide whether or not there exists a control sequence of 0-1 control vectors $\langle u(0), ..., u(M'-1)\rangle$ such that $x(0) = s^{ini}$, $x(M') = s^{tar}$, $M' \leq M$, and the linear cost function $C = \sum_{j=0}^{M'-1}(\sum_{i=1}^{m}(u_i(j) \times g(u_i)))$ is minimum. Then, output one if it exists.

Langmead and Jha proposed an efficient SAT-based method for control of CRBNs [46]. Our proposed method is inspired by this SAT-based method. However, there are some differences between our method and the SAT-based method. Firstly, the SAT-based method is applied to CRBNs while our method is applied to DGARBNs, which are more general than CRBNs. Secondly, the SAT-based method solves the standard control problem while our method solves the optimal control problem, which is more general than the standard control problem. Thirdly, the SAT-based method only supports the time-sensitive mode while our method supports both the time-sensitive and non-time-sensitive modes.

6.1 Time-sensitive mode

For the time-sensitive mode, our intuitive idea is as follows. We first encode an M-length path from x^0 to x^M in the ESTG of the DGARBN as an SMT formula P (11). We then solve P under minimizing the cost function C in Z3 (see [54] for optimization in Z3). If $SAT(P)$, then a control sequence and an optimum cost, which can be easily obtained from the satisfying assignments of the corresponding SMT variables, are released. Otherwise, "there are no control policies" is released.

In (11), T_{start} (8) expresses that the path starts with s^{ini} at time $t = 0$ and T_{end} (10) expresses that the path ends with s^{tar} at time $t = M$. Clearly, we can easily adjust T_{start} and T_{end} to express the case of multiple initial states and/or multiple target states. This is useful since in the biological context we often only consider the values of some dominant genes, other genes can receive arbitrary values. T^M stands for M transitions of this path. Note that $T(x^j, x^{j+1})$ in (9) is a bit different from $T(x^j, x^{j+1})$ in (2). We here replace $f_i(x^j)$ in (2) by $f_i(x^j, u^j)$ where u^j is the vector of values of control nodes at time j.

$$T_{start} \equiv (x_{n+1}^0 = 0) \wedge \bigwedge_{i=1}^{n} (x_i^0 = s_i^{ini}) \tag{8}$$

$$T^M \equiv \bigwedge_{j=0}^{M-1} T(x^j, x^{j+1}) \tag{9}$$

$$T_{end} \equiv \bigwedge_{i=1}^{n} (x_i^M = s_i^{tar}) \tag{10}$$

$$P \equiv T_{start} \wedge T^M \wedge T_{end} \tag{11}$$

Let see a running example on DGARBN (7). In the time-sensitive mode, we obtain a control sequence as $\langle(0,0),(0,0),(0,0),(1,1)\rangle$ and the minimum cost is $C = 3$ for $M = 4$. This result is shown in Table 3. In Table 3, Column "t" denotes the time of evolution (not scaled to LCM), Column "Updated nodes" denoted the nodes which will be updated at time t. When $M = 5$, there are no control sequences. However, in the non-time-sensitive mode, we will obtain the control sequence as for the case $M = 4$.

TABLE 3: The case $M = 4$.

Updated nodes	t	x_1	x_2	x_3	u_1	u_2	cost
x_2, x_3	0	0	0	0	0	0	0
x_1, x_2	1	0	0	0	0	0	0
x_2, x_3	2	1	0	0	0	0	0
x_1, x_2	3	1	0	1	1	1	3
	4	0	1	1			

6.2 Non-time-sensitive mode

Obviously, optimal control of DGARBNs in the non-time-sensitive mode is harder than that in the time-sensitive mode. For standard control of DGARBNs, we can simply find the first target time M_{first} ($0 \leq M_{first} \leq M$) in which the control condition is satisfied (i.e., there exists a control sequence driving the DGARBN from s^{ini} to s^{tar} at time M_{first}). However, for optimal control of DGARBNs, the minimum cost of the case $M = M_{first}$ may not be the smallest minimum cost (the target time corresponding to this smallest minimum cost is called M_{min}, $0 \leq M_{min} \leq M$). Thus, exact methods are needed for optimal control of DGARBNs in the non-time-sensitive mode. We here propose a method to solve this problem.

Based on the method for the time-sensitive mode, we modify the SMT formula (11) to represent an M-length path from x^0 to x^M in the ESTG of the DGARBN such that along with this path, once we reach an extended state satisfying the following condition, all next extended states of the path will be equal to this extended state (i.e., there are no updates). The condition means that the values of internal nodes of the extended state are same as that of the target state s^{tar} and is represented by $\bigwedge_{i=1}^{n}(x_i^j = s_i^{tar})$. The state transition formula of such a path is shown in (12). If the condition does not hold, then we update the DGARBN as usual by the state transition formula T. Otherwise, we do not update the DGARBN. Note that we add a new variable r^j to indicate either the updating case ($r^j = 1$) or the non-updating case ($r^j = 0$). This helps us to represent the new cost function and to easily obtain the real control sequence.

The SMT formula P' of such a path is shown in (14). The cost function is also adjusted as $C' = \sum_{j=0}^{M-1}(\sum_{i=1}^{m}(u_i(j) \times g(u_i) \times r(j)))$ where $r(j)$ corresponds to the variable r^j. We then solve P' under minimizing the cost function C' in Z3. If $SAT(P')$, then a control sequence and an optimum cost are released. Otherwise, "there are no control policies" is released. The optimum cost can be directly obtained from the satisfying assignment of the SMT variable C'. The control sequence can be obtained by first obtaining a sequence of 0-1 control vector $\langle u(0), ..., u(M)\rangle$ from the satisfying assignments of the corresponding SMT variables, and then excluding the spurious vectors. A control vector $u(j)$ is said to be spurious if the satisfying assignment of r^j is 0. Note that M_{min} can be determined as the number of control vectors in the control sequence. Furthermore, if we want to minimize both the cost and the target time, we can simply adjust the cost function, e.g, $C' = \sum_{j=0}^{M-1}(\sum_{i=1}^{m}(u_i(j) \times g(u_i) \times r(j)) \times (M+1) + r(j))$. The adjusted cost function guarantees that the cost is always minimized first, then the target time is minimized. However, since the cost function becomes more complex, the running time may be longer.

$$T'(x^j, x^{j+1}) \equiv \{T(x^j, x^{j+1}) \wedge \neg \bigwedge_{i=1}^{n} (x_i^j = s_i^{tar}) \wedge (r^j = 1)\}$$
$$\vee \{\bigwedge_{i=1}^{n} (x_i^{j+1} = x_i^j) \wedge \bigwedge_{i=1}^{n} (x_i^j = s_i^{tar}) \wedge (r^j = 0)\} \qquad (12)$$

$$T'^{M} \equiv \bigwedge_{j=0}^{M-1} T'(x^j, x^{j+1}) \qquad (13)$$

$$P' \equiv T_{start} \wedge T'^{M} \wedge T_{end} \qquad (14)$$

Let us reconsider the running example in Subsection 6.1 with $M = 5$ and the non-time-sensitive mode. We here change the target state s^{tar} to 111. By applying the proposed method, we can obtain the minimum cost $C' = 2$ and a sequence of control vectors $\langle(0,0),(0,0),(0,1),(0,0),(0,0)\rangle$. We also have $r^0 = 1, r^1 = 1, r^2 = 1, r^3 = 0, r^4 = 0$. By excluding the spurious control vectors, we obtain a control sequence $\langle(0,0),(0,0),(0,1)\rangle$. Herein, $M_{min} = 3$ and is optimal.

6.3 Case study

In this subsection, we apply our method for optimal control of DGARBNs to the apoptosis network, which is very important for programmed cell death and has been widely studied [55]. The BN of this network includes 11 internal nodes and 1 control node. Its Boolean functions are given in Table 4. In this BN, $x_7 = 0$ and $x_9 = 1$ implies cell death while $x_7 = 1$ and $x_9 = 0$ implies cell survival.

TABLE 4: Boolean functions of the BN of the apoptosis network.

Gene	Node	Boolean function
TNF	u	
T2	x_1	$\neg x_8 \wedge u$
IKKa	x_2	$\neg x_6 \wedge \neg x_6 \wedge u$
NF$_K$B	x_3	$\neg x_5$
NF$_K$B$_{nuc}$	x_4	$x_3 \wedge \neg x_5$
I$_K$B	x_5	$(\neg x_2 \wedge x_4 \wedge u) \vee (\neg x_2 \wedge x_4 \wedge \neg u)$
A20a	x_6	$x_4 \wedge u$
IAP	x_7	$(x_4 \wedge \neg x_9 \wedge u) \vee (x_4 \wedge \neg x_9 \wedge \neg u)$
FLIP	x_8	x_4
C3a	x_9	$\neg x_7 \wedge x_{10}$
C8a	x_{10}	$(x_1 \vee x_9) \wedge \neg x_{11}$
CARP	x_{11}	$(x_4 \wedge \neg x_9 \wedge u) \vee (x_4 \wedge \neg x_9 \wedge \neg u)$

Then, we randomly generated four different contexts with $maxP = 3$ for this BN since we do not know the knowledge on the real context of the apoptosis network. We now have four different random DGARBNs. The setting for optimal control of all the four DGARBNs is given as follows. The initial state was set to $(0, ..., 0)$. The values of x_7 and x_9 in the target state were set to 0 and 1, respectively. The other nodes in the target state can receive arbitrary Boolean values. That means the objective of this control is to guide the network toward cell death states. The target time M was set to 50. Since there is only one control node, we simply set $g(u) = 1$. Herein, we consider the non-time-sensitive mode.

Next, we applied our method to the four random DGARBNs. The obtained results are given in Table 5. Column "result" denotes whether a control sequence exists (yes) or not (no). Column "c_{min}" denotes the optimal cost. "-" denotes the case there are no control sequences, c_{min} and M_{min} are undetermined. From these results, we can see that the activation of cell death can be controlled by manipulating the value of the control node.

TABLE 5: Results on optimal control of the apoptosis network.

Network	Result	c_{min}	M_{min}
random-1	yes	2	9
random-2	no	-	-
random-3	yes	1	7
random-4	yes	3	38

7 Experiments

Our methods have been implemented in a JAVA tool integrating the Z3 solver called DABoolNet. An executable file of DABoolNet and some example networks are available at: https://github.com/giang-trinh/daboolnet. To our best knowledge, BooleanNet [25] is the only practical tool for analysis of DA models. However, a DA model is only a special DGARBN (see Subsection 4.1) and BooleanNet is only a simulation tool. Thus, it is difficult to compare DABoolNet with other existing tools with respect to attractor detection and optimal control of DGARBNs. We here only focus on how well our methods scale up.

In order to evaluate the scalability of our methods, we designed two experiments as follows. In the first experiment, we applied our method for attractor detection in DGARBNs to randomly generated DGARBNs. In the second experiment, we applied our method for optimal control of DGARBNs to one artificial example. All experiments were run on a PC with Intel(R) Core(TM) i7 2.40 GHz processor and 16 GB of memory.

7.1 First experiment

In this experiment, we randomly generated 24 DGARBNs with different numbers of nodes and $maxP = 3$. All 24 DGARBNs have the same $LCM = 6$. We then ran DABoolNet to find attractors of these DGARBNs. The time limit was set to four hours for each DGARBN.

Table 6 shows the results of this experiment. Column "n" stands for the number of nodes. Column "$a \times l$" stands for the number and length of attractors computed by DABoolNet. The computational time (in seconds) is given in Column "time". From these results, we see that our method can find attractors of large DGARBNs within practical computational time. Note that since the ESTG of a DGARBN may have $LCM \times 2^n$ extended states, naive approaches (e.g., constructing the ESTG and then applying graph algorithms) are intractable when n is large.

TABLE 6: Results of the first experiment.

n	$a \times l$	time	n	$a \times l$	time
10	2 x 6	2.40	130	16 x 12	2312.64
20	8 x 6	14.49	140	14 x 12	342.61
30	10 x 6	49.37	150	8 x 6, 8 x 12, 4 x 36	711.71
40	16 x 6, 8 x 12	124.72	160	24 x 12	11996.69
50	10 x 12, 2 x 24, 4 x 36	529.95	170	6 x 12	1135.65
60	8 x 24	318.25	180	2 x 12	132.50
70	5 x 12	210.00	190	16 x 12	657.86
80	9 x 12, 2 x 24	324.91	200	4 x 6, 8 x 18, 2 x 36, 2 x 72	2902.37
90	8 x 6	304.19	250	10 x 6, 12 x 30	11338.33
100	4 x 6	136.90	300	5 x 6, 1 x 12, 12 x 30	8290.07
110	3 x 6, 1 x 12	88.64	350	timeout	timeout
120	24 x 12	820.93	400	timeout	timeout

7.2 Second experiment

In this experiment, we consider one artificial example. The artificial DGARBN includes nX nodes and nU control nodes ($nX > nU$). Its Boolean functions are given in Table 7. Its context was randomly generated with $maxP = 3$ (i.e., $LCM \leq 6$). Note that in our proposed methods for optimal control of DGARBNs, the number of variables and the number of clauses of the path formula do not depend on LCM but depend on the number of nodes n and the target time M (see Section 6). Thus, we simply set $maxP$ to 3. The cost vector was set as $G(u_i) = 1 + i\%2$ ($i = 1, ..., nU$). The initial state was fixed to (1,...,1) and the target state was obtained by randomly flipping some nodes in the initial state. In order to evaluate the scalability of our methods for optimal control of DGARBNs, we varied nX, nU, and M. Since optimal control of DGARBNs in the non-time-sensitive mode is harder than that in the time-sensitive mode, we only ran DABoolNet in the non-time-sensitive mode for each combination of nX, nU, and M. Since the time for solving optimal control is usually less than the time for solving attractor detection, the time limit was set to one hour for each combination. Moreover, we also applied two variants of our method for optimal control of DGARBNa in the non-time-sensitive mode (say $\mathcal{M}$) to each combination to compare their performance. Herein, the first variant (say $\mathcal{M}_1$) corresponds to the case of minimizing only the cost while the second variant (say $\mathcal{M}_2$) corresponds to the case of minimizing both the cost and the target time in $\mathcal{M}$.

TABLE 7: An artificial example.

Node	Boolean function
x_1	$x_1 \wedge x_2 \wedge \neg u_1$
x_i ($i = 2, ..., nU$)	$x_{i-1} \wedge x_i \wedge x_{i+1} \wedge \neg u_i$
x_i ($i = nU + 1, ..., nX - 1$)	$x_{i-1} \wedge x_i \wedge x_{i+1}$
x_{nX}	$x_{nX-1} \wedge x_{nX}$

TABLE 8: Results of the second experiment.

				$\mathcal{M}_1$		$\mathcal{M}_2$	
nX	nU	M	result	M_{min}	time	M_{min}	time
200	10	20	no	-	1.04	-	0.98
200	10	40	no	-	2.02	-	2.19
200	10	80	yes	7	37.23	7	37.91
200	20	20	no	-	1.14	-	1.53
200	20	40	yes	1	7.72	1	6.49
200	20	80	no	-	3.69	-	3.87
300	10	20	yes	3	2.66	3	2.70
300	10	40	no	-	2.70	-	3.43
300	10	80	no	-	5.61	-	5.78
300	20	20	yes	10	2.90	10	3.01
300	20	40	yes	23	10.19	5	18.35
300	20	80	yes	7	40.30	1	45.03
400	10	20	yes	19	3.01	4	3.01
400	10	40	yes	5	15.95	5	17.27
400	10	80	yes	4	119.86	4	130.80
400	20	20	no	-	2.08	-	2.11
400	20	40	yes	8	20.17	8	20.85
400	20	80	yes	4	62.58	4	66.23
500	10	20	no	-	2.58	-	2.41
500	10	40	no	-	14.03	-	10.47
500	10	80	no	-	9.52	-	9.60
500	20	20	yes	4	6.87	4	6.34
500	20	40	yes	6	21.93	2	21.83
500	20	80	yes	13	71.98	1	80.78

Table 8 shows the results of this experiment. Column "result" denotes whether a control sequence exists (yes) or not (no). Column "time " stands for the computational time (in seconds) for each combination of nX, nU, and M. "-" denotes the case there are no control sequences, M_{min} is undetermined. In some combinations (e.g., $nX = 300, nU = 20, M = 40$), M_{min} obtained by $\mathcal{M}_2$ is less than M_{min} obtained by $\mathcal{M}_1$. Moreover, $\mathcal{M}_2$ is slower than $\mathcal{M}_1$ in most combinations. These observations are consistent with the analysis on $\mathcal{M}_2$ in Subsection 6.2. In addition, the computational time of $\mathcal{M}_1$ or $\mathcal{M}_2$ for each combination is reasonable even when nX and M are large (e.g., $nX = 500$ and $M = 80$). From these observations, we see that our methods can solve optimal control in the non-time-sensitive mode of large DGARBNs within practical computational time. We also suggest to preferably use $\mathcal{M}_1$ when we only focus on minimizing the cost of applying control nodes and to preferably use $\mathcal{M}_2$ when we focus on minimizing both the cost of applying control nodes and the target time.

8 Conclusion

In this paper, we have proposed the formulation of ESTG. An ESTG captures the whole dynamics of a DGARBN and paves potential ways to analyze and control this DGARBN. Based on this formulation, we have proposed two SMT-based methods for attractor detection and optimal control of DGARBNs, which are two central issues in systems biology. For optimal control of DGARBNs, our method deals with both the time-sensitive mode and the non-time-sensitive mode. Two experiments are designed to evaluate the scalability of our methods. Experimental results show that our methods can be applied to large networks. We have also stated and proved several relations between DGARBNs and other models including DA models, BSBNs, GARBNs, and MxRBNs. These relations not only contribute to understanding the dynamics of random Boolean networks but also pave potential ways to analyze these models based on DGARBNs.

To show the applications of our methods, we have applied our methods to three real biological networks (two networks for attractor detection and one network for optimal control). The results obtained by our methods are consistent with the existing results on these networks in the literature. We have also used our method for attractor detection in DGARBNs to verify some existing numerical insights into the dynamics of CRBNs and DGARBNs.

For attractor detection of DGARBNs, our method suffers an inherent problem of SMT. When LCM of a DGARBN is large (e.g, $LCM \geq 30$), the unfolding depth in our method may be too large even for small DGARBNs. As a consequence, P will have too many variables and clauses, and the time for solving P may be extremely long. To mitigate this problem, further improvement for SMT (e.g., variable ordering) is needed.

Since SMT supports a variety of data types and arithmetic operators, it is potential to extend our methods to that for multi-valued models of DGARBNs where each node can receive multiple values, and more operators are introduced. We also plan to extend our methods to incorporate gene perturbation experiments on DGARBNs.

DGARBNs are particularly useful when information about the kinetics of biological processes is known [14]. However, prior kinetic information is usually not available. In this case, the context (i.e., $p's$ and $q's$) can be randomly sampled from a time interval that is within biological limitations [14], [28]. Additional, MxRBNs are a non-deterministic extension of DGARBNs to deal with the case of lacking knowledge on real contexts. Deterministic asynchronous probabilistic Boolean functions (DA-PBNs) [53] are a stochastic extension of

DGARBNs to deal with the case of lacking information on real Boolean functions. Therefore, proposing efficient methods for attractor detection and optimal control of MxRBNs or DA-PBNs is one of our future work.

References

[1] S. A. Kauffman, *The origins of order: Self-organization and selection in evolution*. OUP USA, 1993.

[2] C. Gershenson, "Introduction to random Boolean networks," *CoRR*, vol. nlin.AO/0408006, 2004. [Online]. Available: http://arxiv.org/abs/nlin.AO/0408006

[3] R. Thomas, "Boolean formalization of genetic control circuits," *Journal of Theoretical Biology*, vol. 42, no. 3, pp. 563–585, 1973.

[4] F. Robert, *Discrete iterations: a metric study*. Springer Science & Business Media, 2012, vol. 6.

[5] C. Gershenson, "Classification of random Boolean networks," in *Proceedings of the Eighth International Conference on Artificial Life*. MIT Press, 2002, pp. 1–8.

[6] M. Noual and E. Goles, "Block-sequential update schedules and Boolean automata circuits," *Discrete Mathematics & Theoretical Computer Science*, 2010.

[7] A. Garg, A. Di Cara, I. Xenarios, L. Mendoza, and G. De Micheli, "Synchronous versus asynchronous modeling of gene regulatory networks," *Bioinformatics*, vol. 24, no. 17, pp. 1917–1925, 2008.

[8] I. Harvey and T. Bossomaier, "Time out of joint: Attractors in asynchronous random Boolean networks," in *Proceedings of the Fourth European Conference on Artificial Life*. MIT Press, Cambridge, 1997, pp. 67–75.

[9] J. A. Papin, T. Hunter, B. O. Palsson, and S. Subramaniam, "Reconstruction of cellular signalling networks and analysis of their properties," *Nature Reviews Molecular Cell Biology*, vol. 6, no. 2, pp. 99–111, 2005.

[10] C. Guet, T. A. Henzinger, C. Igler, T. Petrov, and A. Sezgin, "Transient memory in gene regulation," in *International Conference on Computational Methods in Systems Biology*. Springer, 2019, pp. 155–187.

[11] C. Gershenson, J. Broekaert, and D. Aerts, "Contextual random Boolean networks," in *European Conference on Artificial Life*. Springer, 2003, pp. 615–624.

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

[13] A. Saadatpour, I. Albert, and R. Albert, "Attractor analysis of asynchronous Boolean models of signal transduction networks," *Journal of Theoretical Biology*, vol. 266, no. 4, pp. 641–656, 2010.

[14] P. Bloomingdale, J. Niu, D. E. Mager *et al.*, "Boolean network modeling in systems pharmacology," *Journal of Pharmacokinetics and Pharmacodynamics*, vol. 45, no. 1, pp. 159–180, 2018.

[15] M. Chaves, E. D. Sontag, and R. Albert, "Methods of robustness analysis for Boolean models of gene control networks," *IEE Proceedings-Systems Biology*, vol. 153, no. 4, pp. 154–167, 2006.

[16] E. Goles, M. Montalva, and G. A. Ruz, "Deconstruction and dynamical robustness of regulatory networks: application to the yeast cell cycle networks," *Bulletin of Mathematical Biology*, vol. 75, no. 6, pp. 939–966, 2013.

[17] G. A. Ruz, E. Goles, M. Montalva, and G. B. Fogel, "Dynamical and topological robustness of the mammalian cell cycle network: a reverse engineering approach," *BioSystems*, vol. 115, pp. 23–32, 2014.

[18] F. Greil, B. Drossel, and J. Sattler, "Critical kauffman networks under deterministic asynchronous update," *New Journal of Physics*, vol. 9, no. 10, p. 373, 2007.

[19] C. Grilo and L. Correia, "The influence of asynchronous dynamics in the spatial prisoner's dilemma game," in *International Conference on Simulation of Adaptive Behavior*. Springer, 2008, pp. 362–371.

[20] M. Schneiter, J. Rička, and M. Frenz, "Self-organization of self-clearing beating patterns in an array of locally interacting ciliated cells formulated as an adaptive Boolean network," *Theory in Biosciences*, vol. 139, no. 1, pp. 21–45, 2020.

[21] Z. Li, H. Xiao, and J. Song, "Algebraic approach to asynchronous Boolean networks," in *2011 Chinese Control and Decision Conference (CCDC)*. IEEE, 2011, pp. 769–773.

[22] T. Akutsu, *Algorithms for analysis, inference, and control of Boolean networks*. World Scientific, 2018.

[23] R.-S. Wang, A. Saadatpour, and R. Albert, "Boolean modeling in systems biology: an overview of methodology and applications," *Physical Biology*, vol. 9, no. 5, p. 055001, 2012.

[24] L. De Moura and N. Bjørner, "Z3: An efficient SMT solver," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2008, pp. 337–340.

[25] I. Albert, J. Thakar, S. Li, R. Zhang, and R. Albert, "Boolean network simulations for life scientists," *Source Code for Biology and Medicine*, vol. 3, no. 1, pp. 1–8, 2008.

[26] J. Aracena, J. Demongeot, E. Fanchon, and M. Montalva, "On the number of different dynamics in Boolean networks with deterministic update schedules," *Mathematical Biosciences*, vol. 242, no. 2, pp. 188–194, 2013.

[27] T. Van Giang and K. Hiraishi, "Algorithms for finding attractors of generalized asynchronous random Boolean networks," in *2019 12th Asian Control Conference (ASCC)*. IEEE, 2019, pp. 67–72.

[28] M. Ishii, J. Gores, and C. Teuscher, "On the sparse percolation of damage in finite non-synchronous random Boolean networks," *Physica D: Nonlinear Phenomena*, vol. 398, pp. 84–91, 2019.

[29] S. Huang, "Gene expression profiling, genetic networks, and cellular states: an integrating concept for tumorigenesis and drug discovery," *Journal of Molecular Medicine*, vol. 77, no. 6, pp. 469–480, 1999.

[30] S. Yamanaka, "Elite and stochastic models for induced pluripotent stem cell generation," *Nature*, vol. 460, no. 7251, p. 49, 2009.

[31] K. Tun, M. Menghini, L. D'Andrea, P. Dhar, H. Tanaka, and A. Giuliani, "Why so few drug targets: a mathematical explanation?" *Current Computer-Aided Drug Design*, vol. 7, no. 3, pp. 206–213, 2011.

[32] E. Dubrova and M. Teslenko, "A SAT-based algorithm for finding attractors in synchronous Boolean networks," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 8, no. 5, pp. 1393–1399, 2011.

[33] Q. Yuan, A. Mizera, J. Pang, and H. Qu, "A new decomposition-based method for detecting attractors in synchronous Boolean networks," *Science of Computer Programming*, vol. 180, pp. 18–35, 2019.

[34] A. Mizera, J. Pang, H. Qu, and Q. Yuan, "Taming asynchrony for attractor detection in large Boolean networks," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 16, no. 1, pp. 31–42, 2018.

[35] G. V. Trinh, T. Akutsu, and K. Hiraishi, "An FVS-based approach to attractor detection in asynchronous random Boolean networks," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pp. 1–1, 2020.

[36] C. Barrett, ""Decision procedures: An algorithmic point of view," by Daniel Kroening and Ofer Strichman, Springer-Verlag, 2008," *Journal of Automated Reasoning*, vol. 51, no. 4, pp. 453–456, 2013.

[37] M. Hopfensitz, C. Müssel, M. Maucher, and H. A. Kestler, "Attractors in Boolean networks: a tutorial," *Computational Statistics*, vol. 28, no. 1, pp. 19–36, 2013.

[38] H. Kitano, "Cancer as a robust system: implications for anticancer therapy," *Nature Reviews Cancer*, vol. 4, no. 3, pp. 227–235, 2004.

[39] C. Biane and F. Delaplace, "Causal reasoning on Boolean control networks based on abduction: theory and application to cancer drug discovery," *IEEE/ACM Ttransactions on Computational Biology and Bioinformatics*, vol. 16, no. 5, pp. 1574–1585, 2018.

[40] H. Mandon, C. Su, S. Haar, J. Pang, and L. Paulevé, "Sequential reprogramming of Boolean networks made practical," in *International Conference on Computational Methods in Systems Biology*. Springer, 2019, pp. 3–19.

[41] S. Paul, C. Su, J. Pang, and A. Mizera, "An efficient approach towards the source-target control of Boolean networks," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2019.

[42] C. Su, S. Paul, and J. Pang, "Controlling large Boolean networks with temporary and permanent perturbations," in *International Symposium on Formal Methods*. Springer, 2019, pp. 707–724.

[43] L. C. Fontanals, E. Tonello, and H. Siebert, "Control strategy identification via trap spaces in Boolean networks," in *International Conference on Computational Methods in Systems Biology*. Springer, 2020, pp. 159–175.

[44] C. Su and J. Pang, "Sequential temporary and permanent control of Boolean networks," in *International Conference on Computational Methods in Systems Biology*. Springer, 2020, pp. 234–251.

[45] ——, "A dynamics-based approach for the target control of Boolean networks," in *BCB '20: 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics, Virtual Event, USA, September 21-24, 2020*. ACM, 2020, pp. 50:1–50:8.

[46] C. J. Langmead and S. K. Jha, "Symbolic approaches for finding control strategies in Boolean networks," *Journal of Bioinformatics and Computational Biology*, vol. 7, no. 02, pp. 323–338, 2009.

[47] K. Kobayashi and K. Hiraishi, "Optimal control of Boolean biological networks modeled by Petri nets," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 96, no. 2, pp. 532–539, 2013.

[48] 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.

[49] J. C. Nacher and T. Akutsu, "Minimum dominating set-based methods for analyzing biological networks," *Methods*, vol. 102, pp. 57–63, 2016.

[50] W. Hou, P. Ruan, W.-K. Ching, and T. Akutsu, "On the number of driver nodes for controlling a Boolean network when the targets are restricted to attractors," *Journal of Theoretical Biology*, vol. 463, pp. 1–11, 2019.

[51] S. Lenhart and J. T. Workman, *Optimal control applied to biological models*. CRC press, 2007.

[52] Q. Chai and W. Wang, "A computational method for free terminal time optimal control problem governed by nonlinear time delayed systems," *Applied Mathematical Modelling*, vol. 53, pp. 242–250, 2018.

[53] 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.

[54] N. Bjørner, A.-D. Phan, and L. Fleckenstein, "νZ - an optimizing SMT solver," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2015, pp. 194–199.

[55] 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.

Trinh Van Giang received the B.S. and M.S. degrees in Computer Science from Ho Chi Minh City University of Technology in 2014 and 2017, respectively. He is currently a doctoral student at School of Information Science, Japan Advanced Institute of Science and Technology. His research interests include computational methods for analysis and control of complex dynamical systems.

Kunihiko Hiraishi received from the Tokyo Institute of Technology the B. E. degree in 1983, the M. E. degree in 1985, and D. E. degree in 1990. He is currently a professor at School of Information Science, Japan Advanced Institute of Science and Technology. His research interests include discrete event systems and formal verification. He is a member of the IEEE, IPSJ, and SICE.