ELSEVIER

COMPUTATIONAL ANDSTRUCTURAL BIOTECHNOLOGY JOURNAL

journal homepage: www.elsevier.com/locate/csbj

Mini review

Attractor detection and enumeration algorithms for Boolean networks

Tomoya Mori *, Tatsuya Akutsu *

Bioinformatics Center, Institute for Chemical Research, Kyoto University, Kyoto 611-0011, Japan

ARTICLE INFO

Article history:
Received 25 March 2022
Received in revised form 13 May 2022
Accepted 13 May 2022
Available online 21 May 2022

Keywords:
Boolean network
Singleton attractor
Periodic attractor
Computational complexity
SAT
Nested canalyzing function

ABSTRACT

The Boolean network (BN) is a mathematical model used to represent various biological processes such as gene regulatory networks. The state of a BN is determined from the previous state and eventually reaches a stable state called an attractor. Due to its significance for elucidating the whole system, extensive studies have been conducted on analysis of attractors. However, the problem of detecting an attractor from a given BN has been shown to be NP-hard, and for general BNs, the time complexity of most existing algorithms is not guaranteed to be less than $O(2^n)$. Therefore, the computational difficulty of attractor detection has been a big obstacle for analysis of BNs. This review highlights singleton/periodic attractor detection algorithms that have guaranteed computational complexities less than $O(2^n)$ time for particular classes of BNs under synchronous update in which the maximum indegree is limited to a constant, each Boolean function is AND or OR of literals, or each Boolean function is given as a nested canalyzing function. We also briefly review practically efficient algorithms for the problem.

Contents

* Corresponding authors.
E-mail addresses: tmori@kuicr.kyoto-u.ac.jp (T. Mori), takutsu@kuicr.kyoto-u.ac.jp (T. Akutsu).

https://doi.org/10.1016/j.csbj.2022.05.027

1. Introduction

The *Boolean network* (BN) [1–3] is a mathematical model that is used to represent various biological processes such as gene regulatory networks [4–6], neural networks [7], cell cycle control networks [8], and signal transduction networks [9–12]. For example, when modeling a gene regulatory network, each node corresponds to a gene and is assigned a Boolean value of 0 (FALSE) or 1 (TRUE), which means the gene is inactive or active, respectively. The state of each node is updated according to a Boolean function assigned to it, i.e., the edges between nodes correspond to the regulation relationships between the nodes. In addition, the state of the entire system is called a *global state*, which eventually reaches one of two types of stable states: *singleton attractor* (*point attractor*) or *periodic attractor* (*cyclic attractor*) by repeating the update of the state. Once the global state reaches a singleton attractor, it will always remain in the same state, whereas a periodic attractor consists of multiple global states that are cyclically visited. There is a close relationship between attractors and biological processes. For example, in the case of a gene regulatory network, an attractor is a stable state of the entire system and is considered to be a stable state of a cell. Therefore, one attractor is often regarded as one cell type [3,13,14], and extensive studies have been conducted on attractor detection, due to its significance for elucidating the system. Although it might be pointed out that a BN is a fairly simplified model and it deviates from the actual biological systems, it is meaningful to start the analysis with a simple model in order to decipher a complex system.

A BN of n nodes has 2^n global states. Thus, if we want to detect or enumerate the attractors of a given BN, we can simply update all global states as initial states and find out which initial state reaches which stable state. If n is sufficiently small, this approach is not problematic. When n is large, however, a large amount of computational resources is required. Therefore, algorithms for biological BNs have so far been limited to small networks in most cases [15]. Furthermore, from a theoretical view point, detection of a singleton attractor and enumeration of singleton attractors have been shown to be NP-hard [16,17] and #P-hard [16], respectively. In fact, there is no existing method that works in $o(2^n)$ time (i.e., provably faster than in $O(2^n)$ time) for general BNs. Therefore, algorithms that work efficiently for limited classes of BNs have been proposed [18,19]. For instance, there are some algorithms that detect a singleton attractor in polynomial time for very restricted classes of BNs [20,21] and several $o(2^n)$ time algorithms for reasonably wide classes of BNs [6]. On the other hand, no $o(2^n)$ time algorithm is known for detection of a periodic attractor of period 3 or more for a reasonably wide class of BNs. This may be because detection of an attractor with a long period has been suggested to be PSPACE-hard [6], which is more difficult than problems belonging to class NP unless NP = PSPACE. This theoretical observation also suggests a practical difficulty of detection of a long periodic attractor because PSPACE-hard problems may not be efficiently solved by using practical solvers for NP-hard problems (e.g., SAT-solver, ILP-solver).

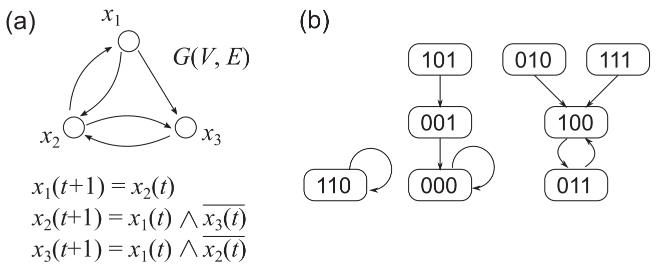
(a)
x_1
$G(V,E)$
x_2
x_3
$x_1(t{+}1) = x_2(t)$
$x_2(t{+}1) = x_1(t) \wedge \overline{x_3(t)}$
$x_3(t{+}1) = x_1(t) \wedge \overline{x_2(t)}$
(b)
101
010
111
001
100
110
000
011

Fig. 1. Example of (a) Boolean network and (b) its state transition diagram.

In this paper, we review $o(2^n)$ time algorithms for singleton/periodic attractor detection for particular classes of BNs under synchronous update in which the maximum indegree is limited to a constant, each Boolean function is AND or OR of literals, or each Boolean function is given as a nested canalyzing function. The reasons why the time complexities of these special classes of BNs have been well-discussed and we focus on them in this paper are (i) each Boolean function can be represented and evaluated in constant space and constant time by limiting the maximum indegree, (ii) it is possible to develop $o(2^n)$ time algorithms by introducing some constraints on types of Boolean functions even when there is no constraint on the maximum indgree, (iii) biologically important functions often have a nested canalyzing form [22–26], and (iv) there are some other BN classes (e.g., *monotonic* BNs [27–29], *monomials* BNs [30,31], *semilattice* BNs [32], BNs whose functions consist of only XOR operations [33], BNs whose functions consist of only AND and NOT operations [34]) but algorithms guaranteed to work in $o(2^n)$ time for the classes have not been proposed as far as we know. We also briefly review practical algorithms for detection and enumeration of attractors for BNs.

2. Boolean network and attractors

A BN $N(V,F)$ consists of a set of nodes $V = \{x_1, x_2, \ldots, x_n\}$ and a set of Boolean functions $F = \{f_1, f_2, \ldots, f_n\}$. Each x_i takes a value of 0 (FALSE) or 1 (TRUE), and in the case of a gene regulatory network, x_i corresponds to the gene i, and 0 and 1 indicate that the gene is inactive or active, respectively. In particular, when writing $x_i(t)$, it indicates the state of the gene i at time t, and the *global state* of the entire system at time t is denoted by $\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]$. A Boolean function $f_i(x_{i_1}, x_{i_2}, \ldots, x_{i_k})$ is assigned to each x_i and it indicates a rule that x_i is controlled by specific input nodes $x_{i_1}, x_{i_2}, \ldots, x_{i_k}$. Here, let $IN(x_i) = \{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}$ be a set of input nodes for x_i. The number of input nodes is denoted by $|IN(x_i)|$ and is called *indegree* of x_i. We use K to denote the maximum indegree for a BN. The state of x_i at time $t+1$ is determined from the state of $IN(x_i)$ at time t according to the corresponding Boolean function $x_i(t+1) = f_i(x_{i_1}(t), x_{i_2}(t), \ldots, x_{i_k}(t))$. This may be simplified and written as $x_i(t+1) = f_i(x_1(t), x_2(t), \ldots, x_n(t))$ if there is no confusion. Furthermore, the global state at time $t+1$ can be written as $\mathbf{x}(t+1) = \mathbf{f}(\mathbf{x}(t))$. The network structure of BN is represented by a directed graph $G(V,E)$, where $E = \left\{\left(x_{i_j}, x_i\right) | x_{i_j} \in IN(x_i)\right\}$. As an example, Fig. 1(a) shows the following BN and the corresponding graph:

$$\begin{aligned} x_1(t+1) &= x_2(t), \\ x_2(t+1) &= x_1(t) \wedge \overline{x_3(t)}, \\ x_3(t+1) &= x_1(t) \wedge \overline{x_2(t)}, \end{aligned}$$

where $x \wedge y$ and $\bar{x}$ denote conjunction (AND) of x and y, and negation (NOT) of x, respectively. Since this BN has three nodes, the number of global states is 2^3 in total. At each time step, the transition of each global state is computed by $\mathbf{x}(t+1) = \mathbf{f}(\mathbf{x}(t))$ and a directed graph representing such transition of the states is called a *state transition diagram* (Fig. 1(b)). As we can see from this state transition diagram, the global state at time $t+1$ is uniquely determined by the global state at time t. For example, if $\mathbf{x}(t) = [1,0,1]$, the global state will be $[0,0,1]$ at time $t+1$ and then $[0,0,0]$ at time $t+2$. However, once it reaches $[0,0,0]$, it remains $[0,0,0]$ no matter how many updates are repeated. This state is called a *singleton attractor* or *point attractor*. Meanwhile, the states $[1,0,0]$ and $[0,1,1]$ repeat the transition between these states and are called a *periodic attractor* or *cyclic attractor*. More generally, an attractor is represented by a set of states $\{\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{p-1}\}$, where $\mathbf{a}_{i+1} = \mathbf{f}(\mathbf{a}_i)$ $(i = 0, 1, \ldots, p-2)$ and $\mathbf{a}_0 = \mathbf{f}(\mathbf{a}_{p-1})$ hold. If $p = 1$, it corresponds to a singleton attractor. If $p > 1$, it corresponds to a periodic attractor and p is called the *period* of the attractor (the more detailed reviews on the attractor can be found in [35]). Since an attractor can be regarded as a stable state of the entire system, detection and enumeration of attractors are important analysis for detailed understanding of the system. Therefore, extensive studies have been done on the distribution and length of attractors. In particular, the *NK model* of BN is often focused on, which consists of N nodes and each node has K input nodes randomly selected and is assigned a Boolean function randomly selected from all possible 2^{2^K} functions. In the analysis of this NK model, it is well known that the expected number of singleton attractors is 1 for every $K > 0$ and $n \geqslant K$ [36,37]. As for the length and distribution of periodic attractors in the NK model, several theoretical studies have been done [38,39].

There exist mainly two types of BNs, *synchronous BNs* whose states of all nodes are updated all at once and *asynchronous BNs* whose states of nodes are updated asynchronously such as *random order asynchronous*, *general asynchronous*, and *deterministic asynchronous* update models [11]. Asynchronous BNs are often employed since the update strategy is reasonable to take into account biological processes which occur in different time scales [6,40]. However, each state of the state transition diagram of an asynchronous BN may have more than one outgoing edges. Therefore, an attractor in an asynchronous BN is often defined as a *strong connected component* (SCC) without outgoing edges (referred to as *bottom SCC* or *terminal SCC*), and the attractor detection is considered as the bottom SCC detection [41]. Such a complex structure of attractor prevents us from applying SAT-based algorithm (described in Section 3.1.1) efficiently since each SAT formula becomes large [42]. Meanwhile, the fact that the synchronous model and the general asynchronous model have the same set of singleton attractors can be shown [6]. Although we deal with synchronous BNs in this paper unless otherwise stated since extensive studies on attractor detection have been done due to their simplicity, some practical attractor detection algorithms for asynchronous BNs are described in Section 5. In addition to synchronous and asynchronous BNs, another semantics on BNs called *most permissive* has been proposed and studied [43]. It was shown in [43] under this semantics that singleton attractor detection (i.e., detection of a fixed-point) remains NP-complete but reachability and attractor membership problems can be solved better than in PSPACE (under reasonable assumptions on the complexity classes).

3. Detection and enumeration of singleton attractors

One of the simplest methods for detecting or enumerating attractors is to repeat the state transitions with all possible global states as initial states for a given BN and to find out which state each initial state finally reaches. This task can be done by constructing and analyzing the state transition diagram. However, the state transition diagram for a BN with n nodes consists of 2^n nodes and 2^n edges. Therefore, if n is large, it is impossible to construct and analyze the state transition diagram. In fact, the singleton attractor detection problem has been shown to be an NP-hard problem by using a reduction from 3-SAT (Boolean satisfiability problem) [6,17]. Thus, various algorithms that work efficiently for specific classes of BNs have been proposed. In this section, we introduce singleton attractor detection/enumeration algorithms for BNs with maximum indegree K, BNs in which each Boolean function is AND or OR of literals, and BNs configured by nested canalyzing functions.

3.1. Singleton attractor detection for BNs with maximum indegree K

This section deals with BNs with maximum indegree K, where K is any constant larger than 1. Since the number of states of K input nodes is 2^K and both 0 and 1 can be assigned as an output value for each of such states, there exist 2^{2^K} Boolean functions with K inputs. That is, if K is a constant, each Boolean function is specified by a constant space and can be evaluated in constant time.

3.1.1. Reduction to SAT

We have already mentioned that SAT is used to show the NP-hardness of the attractor detection problem, but it can also be applied to the singleton attractor detection itself [44–46]. More specifically, the singleton attractor detection for BNs with maximum indegree K can be replaced by the $(K+1)$-SAT of n variables. SAT is a problem of determining whether there is a $0-1$ vector (*assignment*) $\mathbf{a}$ that satisfies $f(\mathbf{a}) = 1$ when a Boolean function f is given by the following *conjunctive normal form* (CNF),

$$C(\mathbf{x}) = \bigwedge_i \bigvee_j \ell_{i,j},$$

where $\ell_{i,j}$ is the jth literal of the ith clause. Note that ℓ is called a *literal* if it is a Boolean variable or its negation, and OR of literals is called a *clause*. Especially when each clause consists of at most k literals, it is called k-SAT. For example, for a BN with $K = 2$, we consider the following transformations for the two-variable Boolean operations, $\wedge$ (conjunction, AND), $\vee$ (disjunction, OR), and $\oplus$ (exclusive OR, XOR) (the cases of constant and unary functions are omitted since they are trivial).

$$\begin{aligned} x_i = \ell_j \wedge \ell_k \Longleftrightarrow \quad & (\overline{x_i} \vee \ell_j \vee \ell_k) \wedge (x_i \vee \overline{\ell_j}) \wedge (x_i \vee \overline{\ell_k}), \\ x_i = \ell_j \vee \ell_k \Longleftrightarrow \quad & (\overline{x_i} \vee \ell_j) \wedge (\overline{x_i} \vee \ell_k) \wedge (x_i \vee \overline{\ell_j} \vee \overline{\ell_k}), \\ x_i = \ell_j \oplus \ell_k \Longleftrightarrow \quad & (\overline{x_i} \vee \ell_j \vee \ell_k) \wedge (\overline{x_i} \vee \overline{\ell_j} \vee \overline{\ell_k}) \wedge (x_i \vee \overline{\ell_j} \vee \ell_k) \wedge (x_i \vee \ell_j \vee \overline{\ell_k})), \end{aligned}$$

where, in a singleton attractor, $x_i(t+1) = x_i(t)$ holds for all $i = 1, \ldots, n$, so t is omitted to express the Boolean variables. This means that each Boolean function can be converted to a 3-SAT consisting of at most four clauses. Furthermore, by applying this transformation to all Boolean functions assigned to x_i $(i = 1, \ldots, n)$ and then combining them with $\wedge$, the singleton attractor detection problem of a BN with n nodes and $K = 2$ can be converted into a 3-SAT problem consisting of n variables and at most $4n$ clauses. In general, the singleton attractor detection for a BN with n variables and the maximum indegree K can be converted in polynomial time to $(K+1)$-SAT with at most $2^{K+1} \cdot n$ clauses and n variables. Although readers may think it is strange to use reduction to $(K+1)$-SAT (not to K-SAT), it is reasonable because 2-SAT is solvable in polynomial time whereas singleton attractor detection for $K = 2$ is NP-hard [6]. To actually find a singleton attractor, we need to apply an efficient SAT solver after converting to SAT by the above method. Although SAT is an NP-hard problem, various practical solvers were developed which can solve large-scale SAT instances [47]. Furthermore, various theoretically efficient algorithms have also been developed. For example, an $O(1.3303^n)$ time algorithm is known for 3-SAT [48], which implies that the singleton attractor detection problem can be solved in $O(1.3303^n \cdot poly(n))$ time for BNs with $K = 2$, where $poly(n)$ means some polynomial function of n.

3.1.2. Using feedback vertex set

The singleton attractor detection can also be done by using the *feedback vertex set* (FVS) in graph theory [16]. The FVS is a subset of nodes $U \subseteq V$ in a directed graph $G(V,E)$ such that removal of all incoming edges to U eliminates all directed cycles (Fig. 2). In particular, the FVS with the smallest number of nodes is called the minimum FVS. If the state of each node $v \in U$ included in FVS is fixed, the state of $IN(v)$ is irrelevant to the update of v, so that, the state of FVS is propagated to all nodes except FVS in at most $n-1$ steps in a graph G with n nodes. In other words, by fixing the state of FVS of a directed graph $G(V,E)$ corresponding to a given BN $N(V,F)$, the BN reaches a stable state in at most $n-1$ steps. However, it should be noted that not all stable states correspond to singleton attractors. For example, if FVS $= \{v_1, v_2, v_3\}$ and $v_1(t+1) = v_2(t) \wedge v_3(t)$, BN reaches a specific stable state by fixing $[v_1, v_2, v_3] = [0,1,1]$, but this state is not a singleton attractor for a given BN since v_1 must be 1. Therefore, in order to enumerate singleton attractors, it is enough to assign all possible states to the minimum FVS and to update the states until the BN reaches stable states, and then to check their consistencies. Assuming that the size of minimum FVS is $|U|$ and each Boolean function of $N(V,F)$ can be evaluated in polynomial time, enumeration of all singleton attractors can be done in $O\left(2^{|U|} \cdot poly(n)\right)$ time since the number of possible states of FVS is $2^{|U|}$. It is known that the problem of finding an FVS is NP-hard, but some practical algorithms have been proposed [49–51], and they can be used for finding singleton attractors. The FVS-based algorithm has been extended for enumeration of periodic attractors in BNs [6] and for more general non-linear models of biological networks [52]. In addition, Mori and Mochizuki also reported the relationship between the expected number of singleton attractors and the feedback arc sets [53].

3.1.3. Recursive algorithms

Here, we introduce a simple recursive algorithm having a guaranteed average-case complexity for singleton attractor enumeration [54]. This recursive algorithm considers a partial global state $[x_1, \ldots, x_m]$ for $m < n$, and iteratively examines whether there is a contradiction between the partial global states and $f_i(\mathbf{x})$ assigned to x_i while incrementing m. If there is no contradiction between all x_i and $f_i(\mathbf{x})$, output it as a singleton attractor, otherwise stop examining the current assignment and proceed to the next assignment. For example, consider the BN shown in Fig. 1(a) as an example and give $\mathbf{x} = [\epsilon, \epsilon, \epsilon]$ as an initial state, where ϵ represents a state that has not yet been determined. Then, we examine the partial global state of $\mathbf{x} = [0, \epsilon, \epsilon]$. At this time, x_2 and x_3 are undetermined, thus $\mathbf{f}(\mathbf{x})$ is consistent. Therefore, we expand the subset of nodes to be assigned and examine $\mathbf{x} = [0, 0, \epsilon]$. Since there is no contradiction in $\mathbf{f}(\mathbf{x})$ as well, we further expand the subset and examine $\mathbf{x} = [0,0,0]$. Here, $\mathbf{f}(\mathbf{x})$ is consistent and the states of all nodes are determined. Therefore, $\mathbf{x} = [0,0,0]$ is reported as a singleton attractor. In the next step, $\mathbf{x} = [0,0,1]$ is examined. Since $f_3(\mathbf{x}) \neq x_3$, $\mathbf{x}$ is not a singleton attractor and nothing is output. Then, we go back to the previous recursion level and examine $\mathbf{x} = [0, 1, \epsilon]$. In this case, although x_3 is not yet determined, $f_1(\mathbf{x}) \neq x_1$ is already established, so that the examination of $[0, 1, \epsilon]$ is stopped and then $\mathbf{x} = [1, \epsilon, \epsilon]$ is examined. By repeating this procedure, it is possible to list all singleton attractors. Letting $s = \frac{m}{n}$, theoretical analysis shows that this algorithm works in $O\left(\left(\max_s (2 - s^K)^s\right)^n \cdot poly(n)\right)$ time on the average, where the average is taken over all NK models, and $0 < s < 1$. For example, the resulting time complexity is $O(1.35^n)$ and $O(1.67^n)$ for $K = 2$ and $K = 10$, respectively. Therefore, when K is small, it is sufficiently faster than $O(2^n)$. In this algorithm, the partial global states are determined in a given order of $x_1, \ldots, x_n$. However, if the nodes are sorted in advance in the descending order of their outdegrees, the number of nodes whose states are determined at each recursive step will increase, so that it is expected that the singleton attractor can be detected with a smaller number of trials. Actually, the algorithm based on this idea has been shown to work in $O(1.19^n)$ time and $O(1.57^n)$ time for $K = 2$ and $K = 10$, respectively. Furthermore, this algorithm can be extended for enumeration of periodic attractors with a small fixed period p [54].

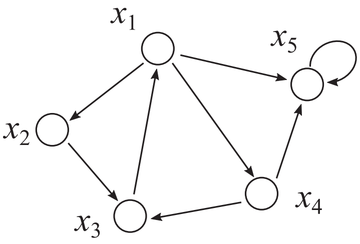

Fig. 2. Example of a BN for illustration of FVS. $\{x_1, x_5\}$ is the minimum FVS. $\{x_2, x_4, x_5\}$ is a FVS but is not the minimum one.

3.2. Singleton attractor detection for AND/OR BNs

This section considers AND/OR BNs, in which each Boolean function is AND or OR of literals. For singleton attractor detection for AND/OR BNs, there is an algorithm that works in $O(1.792^n)$ time, which is a combination of recursive calls and existing methods for solving SAT [55]. To explain the algorithm, first we assume that the following BN function is assigned for x_i.

$$x_i = x_1 \wedge x_2 \wedge \ldots \wedge x_h.$$

In this case, the consistent $0-1$ assignments for $[x_1, x_i]$ are $[0,0], [1,0]$, and $[1,1]$, whereas $[0,1]$ is inconsistent since x_i must be 0 if x_1 is 0, so that the assignment for $[x_1, x_i] = [0,1]$ does not need to be considered anymore. Thus a singleton attractor may be obtained by examining the other three assignments and repeating the same procedure for the other nodes. Since only 3 among $2^2 = 4$ assignments are examined per two nodes, this might lead to an $O\left(3^{n/2} \cdot poly(n)\right) = O(1.733^n \cdot poly(n))$ time algorithm. However, this procedure cannot be continued if there are no remaining edges or only nodes with self-loops. Let U be the set of nodes whose $0-1$ assignments are already determined by this procedure. In such case, the assignments of the set of remaining nodes W (i.e., $W = V \setminus U$) are determined by the following procedure.

- if $|U| > \alpha n$, examine all possible assignments on W,
- otherwise, compute consistent assignments on W using a SAT algorithm.

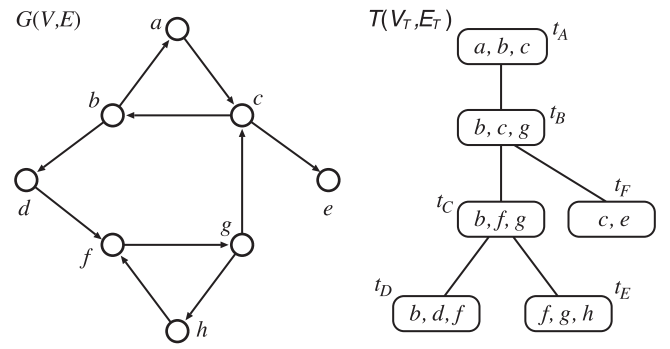

Fig. 3. Example of graph $G(V,E)$ and its tree decomposition $\left\langle \mathscr{T}(\mathscr{V}_{\mathscr{T}},\mathscr{E}_{\mathscr{T}}),(B_t)_{t\in\mathscr{V}_{\mathscr{T}}}\right\rangle$ with width two, where $t_A,\ldots,t_F$ are bags.

It is shown that by letting $\alpha = 0.767$ and utilizing an $O(1.234^m \cdot poly(n))$ time algorithm for SAT with m clauses [56], the resulting algorithm works in $O(1.792^n)$ time. By further omitting unnecessary examinations of partial assignments, an $O(1.587^n)$ time algorithm was developed [57]. It is to be noted that these results might be slightly improved by using a recent $O(1.223^m \cdot poly(n))$ time algorithm for SAT with m clauses [58].

3.3. Singleton attractor detection for nested canalyzing BNs

A Boolean function f_v (assigned to a node v) is called *nested canalyzing* if it has the following form:

$$f_v = \ell_1 \vee \cdots \vee \ell_{k_1-1} \vee \left(\ell_{k_1} \wedge \cdots \wedge \ell_{k_2-1} \wedge \left(\ell_{k_2} \vee \cdots \vee \ell_{k_3-1} \vee (\cdots)\right)\right),$$

where $\ell_i \in \{x_1, \overline{x_1}, x_2, \overline{x_2}, \ldots, \overline{x_n}\}$ and $1 \leqslant k_1 < k_2 < \cdots$. A BN is called an *nc-BN* if nested canalyzing functions are assigned to all nodes. It should be noted that both AND and OR functions of literals are special cases of nested canalyzing functions. Since biologically important functions often have a nested canalyzing form [22–26], it is meaningful to consider attractor detection algorithms for nc-BNs.

3.3.1. Recursive algorithm for singleton attractor detection for nc-BNs

Here, we briefly introduce the basic idea of a recursive algorithm **SattNC** for singleton attractor detection for nc-BNs [19].

The first OR part of f_v (i.e., $\ell_1 \vee \cdots \vee \ell_{k_1-1}$) is called the *initial clause*. Suppose that u appears positively in the initial clause of some f_v where $u \neq v$. If 1 is assigned to u, then we have $f_v = 1$ and thus the state of v becomes 1 regardless of the state of other input nodes to v. This means that assigning 1 to v determines the states of two nodes (u and v). On the other hand, if 0 is assigned to u, the state of v may not be determined. Conversely, suppose that u appears negatively in f_v (i.e., $f_v = \overline{u} \vee f_{v'}$). In this case, $u = 0$ determines the states of two nodes (u and v), whereas $u = 1$ determines the state of one node (u). This procedure may be applied recursively. Let $G(q)$ be the number of recursive calls with q unassigned variables in such a recursive procedure. Then, we may have

$$G(q) \leqslant G(q-1) + G(q-2)$$

from the above discussion. If we can repeat this procedure until the states of all nodes are determined, $G(q)$ will be $O(1.619^q)$ (the Fibonnaci number). However, in many cases, this procedure cannot be repeated so many times. Furthermore, assigning $u = 1$ (resp., $u = 0$) gives a constraint $f_u = 1$ (resp., $f_u = 0$). In order to cope with these issues, we need to solve SAT for a set of nested canalyzing functions, which is more general than the standard SAT for clauses (i.e., a set of OR of literals). **SattNC** was obtained by combining the above recursive procedure with a newly developed SAT algorithm for nested canalyzing functions, and was shown to work in $O(1.871^n)$ time [19].

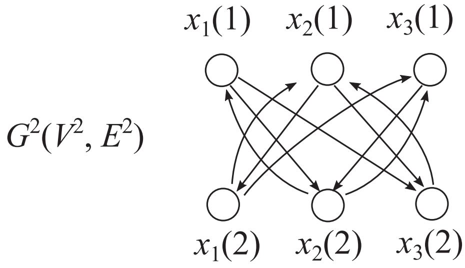

Fig. 4. p-multiplied network $G^2 = \left(V^2, E^2\right)$ corresponding to $G(V,E)$ given in Fig. 1 (a).

3.3.2. Singleton attractor detection for nc-BNs with bounded treewidth

It is known that many NP-hard problems can be solved in polynomial time using dynamic programming if an input graph has a fixed *treewidth*, where the treewidth is an integer value measuring how close a graph is to a tree [59,60]. To formally define the treewidth, we consider the *tree decomposition* that transforms $G(V,E)$ to a pair $\left\langle \mathscr{T}(\mathscr{V}_{\mathscr{T}},\mathscr{E}_{\mathscr{T}}),(B_t)_{t\in\mathscr{V}_{\mathscr{T}}}\right\rangle$, where $\mathscr{T}(\mathscr{V}_{\mathscr{T}},\mathscr{E}_{\mathscr{T}})$ is a rooted tree, $(B_t)_{t\in\mathscr{V}_{\mathscr{T}}}$ is a family of subsets of nodes $V = \{v_1, \ldots, v_n\}$, each B_t is called a *bag*, each node $v_i \in V$ must appear in at least one B_t, nodes in each edge $e \in E$ must be included in at least one B_t, and B_t's containing each node must be connected in $\mathscr{T}$ (see Fig. 3). Note that the tree decomposition is not uniquely determined. The width of the decomposition is defined by $\max_{t\in\mathscr{V}_{\mathscr{T}}}(|B_t| - 1)$and the treewidth of G is the smallest width among all tree decompositions of G.

It is known that the singleton attractor detection problem can be solved in $O(n^{2(w+1)} \cdot poly(n))$ if a given BN $N(V,F)$ is composed of nested canalyzing functions and the treewidth of $G(V,E)$ is bounded by w [61]. This algorithm applies dynamic programming to $\mathscr{T}(\mathscr{V}_{\mathscr{T}},\mathscr{E}_{\mathscr{T}})$ and computes partial assignments from leaves to the root in a bottom-up manner. Chang et al. have improved this algorithm for the special cases of AND/OR BNs and nc-BNs [62]. There is also an algorithm that reduces singleton attractor detection for nc-BNs with bounded treewidth to a *constraint satisfaction problem* for bounded treewidth [61,63].

4. Detection of periodic attractors

As mentioned in Section 1, detection of a periodic attractor seems much harder than detection of a singleton attractor. Nevertheless, some $o(2^n)$ time algorithms have been developed for detection of periodic attractors with a short period. This section briefly introduces such algorithms.

4.1. Reduction to the singleton attractor detection

A simple strategy for detecting periodic attractors is to reduce it to the singleton attractor detection problem. Here, we introduce a simple reduction algorithm that constructs *p-multiplied network* $N^p(V^p, F^p)$ from an input BN $N(V,F)$ [61]. $N^p(V^p, F^p)$ is defined as follows:

$$\begin{aligned}
V^p = \;& \left\{x_i^{(1)}, x_i^{(2)}, \cdots, x_i^{(p)} | x_i \in V\right\}, \\
F^p = \;& \left\{f_i^{(1)}\left(x_{i_1}^{(p)}, \cdots, x_{i_{k_i}}^{(p)}\right), \right. \\
& f_i^{(2)}\left(x_{i_1}^{(1)}, \cdots, x_{i_{k_i}}^{(1)}\right), \cdots, \\
& f_i^{(p)}\left(x_{i_1}^{(p-1)}, \cdots, x_{i_{k_i}}^{(p-1)}\right). \\
& \left. | f_i\left(x_{i_1}, \ldots, x_{i_{k_i}}\right) \in F \mathrm{and} f_{i(t)} \cong f_i\right\},
\end{aligned}$$

where each $x_i^{(t)}$ is regarded as a node and $f_i^{(t)} \cong f_i$ means that $f_i^{(t)}$ is the same Boolean function as f_i except that input variables are different (see Fig. 4). A singleton attractor at $N^p(V^p, F^p)$ clearly corresponds to an attractor of $N(V, F)$ with period q that divides p. However, in order to guarantee that the detected attractor has exactly period p, for all $q = 2, \ldots, p$, $x_i^{(1)} \neq x_i^{(q)}$ must hold for some $x_i \in V$. This means that the state of $N(V, F)$ at time $t = 1$ must be different from that of at time $t = 2, \ldots, p$, so that $N(V, F)$ does not take the same global state at $t = q_1$ and $t = q_2$ $(1 \leqslant q_1 < q_2 \leqslant p)$. Here, let $\phi(t)$ be a function from $\{2, \ldots, p\}$ to $\{1, \ldots, n\}$ and $\psi(t)$ be a function from $\{2, \ldots, p\}$ to $\{0, 1\}$. To check if there is a periodic attractor with period p of $N(V, F)$, it is enough to check whether there exist an singleton attractor of $N^p(V^p, F^p)$ and functions $\phi(t)$ and $\psi(t)$ such that $x_{\phi(t)}^{(1)} = \psi(t)$ and $x_{\phi(t)}^{(t)} = 1 - \psi(t)$ holds for all $t = 2, \ldots, p$. The possible number of ϕs and ψs are n^{p-1} and 2^{p-1}, respectively, which are in polynomial of n for a constant p. Therefore, if singleton attractor detection can be done in $O((1+\delta)^n)$ time, p-periodic attractor detection canbe done in $O((1+\delta)^{pn} \cdot poly(n))$ time. However, even if the $O(1.587^n)$ time algorithm for singleton attractor detection for AND/OR BNs is applied, it does not yield an $o(2^n)$ time algorithm because it would take $O\left(1.587^{2n} \cdot poly(n)\right)$ time ($1.587^2 \approx 2.519$) to detect a periodic attractor with period 2.

4.2. 2-periodic attractor detection for AND/OR BNs

This section describes a 2-periodic attractor detection algorithm for AND/OR BNs using $N^2\left(V^2, F^2\right)$ [61]. First, we transform an AND/OR BN to an OR BN. Let x_i be an AND node assigned the following function:

$$x_i(t+1) = \ell_{i_1} \wedge \ldots \wedge \ell_{i_k}.$$

This Boolean function can be transformed to an OR function by replacing it to

$$x_i(t+1) = \overline{\ell_{i_1}} \vee \ldots \vee \overline{\ell_{i_k}}$$

and negating all $x_i(t+1)$ in the function f_j for all $x_j \in V$. Therefore, in this section, we can assume that $N^2\left(V^2, F^2\right)$ is an OR BN.

The basic strategy of the 2-periodic attractor detection algorithm is similar to that of the $O(1.587^n)$ time algorithm for an AND/OR BN, that is, we recursively examine $0-1$ assignments and finally apply an SAT algorithm. However, here we use the special property on $N^2\left(V^2, F^2\right)$. The procedure of 2-periodic attractor detection is to first construct $N^2\left(V^2, F^2\right)$ and set all nodes to be unassigned. Then, we recursively examine $0-1$ assignments for unassigned nodes x such that $U(x) \geqslant 3$ until no such nodes exist or the number of assigned nodes is more than H, where $U(x)$ is the number of unassigned neighboring nodes of x and H is a parameter. Here, let A be the set of assigned nodes, and let $A_1 = A \cap V_1^2$ and $A_2 = A \cap V_2^2$ (w.l.o.g., $|A_1| \geqslant |A_2|$), where V_1^2 and V_2^2 are the sets of nodes V^2 corresponding to $t = 1$ and $t = 2$, respectively. If $|A| > H$, we examine all possible assignments for $V_1 \setminus A_1$, otherwise recursively examine assignments for paths and cycles (because all nodes with $U(x) \geqslant 3$ have already been assigned), and then finally solve SAT. The resulting algorithm has been shown to work in $O(1.985^n)$ time by letting $H = 0.3196n$ [61]. It is known that detection of 2-periodic attractor can be done in linear time for a positive OR BN in which all Boolean functions are OR functions and all variables appear positively [21,61]. However, periodic attractor detection remains NP-hard for a positive BN in which each function is an AND or OR function [64].

4.3. Periodic attractor detection for nc-BNs with bounded treewidth

The algorithm presented in Section 3.3.2 can be extended for detection of a p-periodic attractor for an nc-BN with bounded treewidth, using the p-multiplied network $N^p(V^p, F^p)$ [61]. The extended algorithm is based on the following proposition: if the graph $G(V, E)$ associated with $N(V, F)$ has the treewidth w, the graph $G^p(V^p, F^p)$ associated with $N^p(V^p, F^p)$ has a tree decomposition with the width less than $p(w+1)$, and for each $x \in V$, $x^{(1)}, \ldots, x^{(p)}$ are included in the same B_t [61]. Then, we can apply a dynamic programming procedure to the tree decomposition of $G^p(V^p, F^p)$ associated with $N^p(V^p, F^p)$, where some modifications are needed to detect an attractor with exactly period p. The resulting algorithm works in $O(n^{2p(w+1)} \cdot poly(n))$ time [61]. Furthermore, the algorithm works in $O(g(d, p, w) \cdot poly(n))$ time if the maximum indegree is bounded by a constant d where g is a non-polynomial function depending on d, p, w but not depending on n. It gives a fixed-parameter algorithm when p, w, and d are parameters (i.e., the exponential factor of the time complexity depends only on some parameters for the algorithm) [59,60].

5. Practical algorithms

In the previous sections, we have introduced singleton/periodic attractor detection and enumeration algorithms that are guaranteed to work in less than $O(2^n)$ time. However, these algorithms are limited to particular subclasses of BNs and no such an algorithm has been known for general BNs. On the other hand, there are many algorithms that do not have such a theoretical guarantee but work efficiently in practice.

5.1. SAT and logic-based approaches

One major practical approach is to use SAT and related logic-based methods. As mentioned in Section 3.1.1, attractor detection problems can be reduced to SAT. Furthermore, many practically efficient SAT solvers have been developed [44,45]. Therefore, it is reasonable to develop practically efficient attractor detection/enumeration methods for BNs and related models using such SAT solvers. For example, Dubrova and Teslenko used a SAT solver to search for a path of length p on the state transition diagram and detected an attractor by checking if it contains a loop [45], and de Jong and Page transformed the problem of searching for a stable state of a network described by the pairwise-linear differentiation equation model into SAT [44]. For other logic-based approaches, Devloo et al. developed an algorithm applying constraint programming to the singleton attractor detection and enumeration [65]. Inoue provided an algorithm that directly encodes a BN into a logic program and computes a singleton attractor based on that logic program [66], and Abdallah et al. proposed an algorithm based on *answer set programming* (ASP) that enumerates all attractors without creating an entire state transition diagram [67]. In addition to SAT, *Binary decision diagrams* (BDDs) have also been utilized to solve large scale logic-based problems in various fields. Therefore, it is reasonable to apply BDDs in place of SAT solvers. Indeed, some methods have been developed based on BDDs that can detect attractors for large networks [68–70]. Integer linear programming (ILP) is another useful method to efficiently solve Boolean constraints and thus has been applied to attractor detection and related problems on BNs [71–73].

5.2. Network reduction-based approaches

Another major practical approach is to use network reduction. For example, suppose that v has an input node u and u has only one input node w. Then, for both detection and enumeration of singleton attractors, we can remove u by letting w as an input to v. Although this example is a very simple one, various reduction methods were developed where logic-based or other methods were used to solve reduced instances at the final stage. For example, Veliz-Cuba et al. proposed various reduction rules including the above mentioned one, elimination of redundant edges, replacement of functions, and simplification of Boolean functions [74]. Veliz-Cuba et al. developed a singleton attractor enumeration algorithm that transforms input networks to AND-NOT networks, then applies network reduction, and finally applies an algebraic method (compute the Gröbner basis to perform a generalized version of Gaussian elimination) [75]. He et al. developed a reduction method based on removal of unstable partial states and identification of constant nodes [76]. Saadatpour et al. provided rigorous proofs of conservation of attractors in synchronous and asynchronous BNs for various reduction methods [77]. Beneš et al. proposed another network reduction method called *interleaved transition guided reduction* (ITGR) for detecting of bottom SCCs of asynchronous BNs [78]. They succeeded in handling large Boolean networks of real data (up to 350 variables) as well as of synthetic data (up to 1100 variables) based on the reduction technique. Gao et al. characterized periodic attractors of a conjunctive BN (i.e., each Boolean function is AND of literals) whose underlying directed graph is strongly connected by establishing bijection between the set of periodic attractors and the set of binary necklaces (i.e., character strings over the binary set $\{0,1\}$, where their all rotations are dealt with as equivalent) of a certain length [79]. Chen et al. extended this study to BNs over weakly connected directed graph by applying network reduction [80]. It is to be noted that network reduction methods were utilized also in some other methods mentioned in this section.

5.3. Network decomposition-based approaches

Divide-and-conquer is one of the major general techniques to efficiently solve various combinatorial problems in computer science. Therefore, it is reasonable to apply divide-and-conquer to the attractor detection/enumeration problems on BNs. Indeed, various methods have been developed for the problem by decomposing the original network into subnetworks and then reconstructing global attractors by integrating local attractors for the subnetworks, where the techniques introduced in Sections 5.1 and 5.2 are also utilized in many of them. As a pioneering work in this direction, Irons proposed an algorithm that combines partial state and predecessors [81]. Mizera et al. developed another algorithm by using decomposition of the original BN into *strongly-connected components* (SCCs) [41], where an SCC is a well-known concept in graph theory and is a maximal subnetwork in which all node pairs are connected by directed paths. Su et al. significantly improved this algorithm by further partitioning SCCs [82]. Zañudo and Albert introduced the concept of a *stable motif*, which is an SCC that stabilizes in attractors and made use of stable motifs to efficiently find attractors in the whole BN [83]. Klarner et al. introduced a similar concept, *trap space*, and combined it with ILP [72] and model checking [84]. Choo and Cho developed a method based on hierarchically partitioning with focusing on attractors corresponding to a particular phenotype of interest instead of considering all attractors [85]. Tamaki developed an algorithm combining *path-decomposition* and partial states [86]. It should be noted that the *tree decomposition*-based methods mentioned in Sections 3.3.2 and 4.3 can also be regarded as network decomposition-based ones.

5.4. FVS-based approaches

The FVS-based singleton attractor detections for synchronous BNs are described in Section 3.1.2. Meanwhile, practical FVS-based approaches for asynchronous random Boolean networks (ARBNs) have also been proposed [42,87]. Skodawessely and Klemm focused on a *reduced dynamics* to make it easier to find attractors of ARBNs [87]. A reduced dynamics is obtained by retaining a Boolean state of a node in FVS and then removing state transitions from the state transition diagram. Finally, the attractors of an ARBN can be found from the sets of fixed points of the reduced dynamics. Although this approach seems to be widely applicable, it is often difficult to handle large networks ($n \geqslant 30$) since it requires large memory space to store all the state vectors of an attractor during the traverse of the original state transition diagram. Van Giang et al. were inspired by the concept of reduced dynamics and presented the relations between FVSs and dynamics of BNs with formal proofs [42]. Furthermore, they proposed another FVS-based method for detecting attractors in ARBNs based on the relations. Its main idea is to systematically remove edges from the transition diagram and then filter out the set of fixed points of them to get a candidate set of states which one-to-one corresponds to the set of attractors of the given ARBN. The computational experiments using real biological networks and random *N-K* networks showed that the method succeeded in handling large networks whose sizes are up to 101 without any network reductions.

5.5. Other approaches

It is well-known that starting from an arbitrary state and repeating update of states, the trajectory will finally fall into a singleton or periodic attractor. Therefore, starting from many initial states, we may be able to get directed attractors or some statistics on attractor distributions. Although such sampling-based studies have been conducted on small-size BNs, it is difficult to apply such methods to analysis of large-scale BNs.

Another major approach is to use *semi-tensor product* (STP) [5], which is an extension of matrix product. In this approach, many problems on BNs are defined as matrix-based problems. Since matrices play central roles in control theory, various concepts and methodologies in control theory have been applied to BNs and thus many studies have been done using STP [88]. For example, a state transition diagram can be represented as a binary matrix A of size $2^n \times 2^n$ in which $A_{ij} = 1$if and only if there exists a transition from state i to state j [5]. Then, singleton attractors correspond to elements of A such that $A_{ii} = 1$. Therefore, once A is constructed, detection and enumeration of singleton attractors become trivial [5]. Although most of existing STP-based methods need to handle $2^n \times 2^n$ or larger size matrices and thus can only handle small-size BNs, some efforts have been done to address this complexity issue [89,90].

6. Conclusion

In this article, we briefly reviewed algorithms for singleton/periodic attractor detection/enumeration with focusing on synchronous BNs. Enumeration of all singleton and periodic attractors can trivially be done by constructing and analyzing a state transition diagram. However, such an approach needs $O(2^n \cdot poly(n))$ or more computation time because there exist 2^n possible states. Since 2^n is often too large, it is important to develop theoretically and/or practically efficient algorithms. However, it is not an easy task because it is known that detection of a singleton attractor is NP-hard, enumeration of a singleton attractor is #P-hard, and detection of a long periodic attractor is suggested to be PSPACE-hard [6].

We have seen that singleton attractor detection can be done in provably faster than in $O(2^n)$ time (i.e., still exponential, but $o(2^n)$ time) for some special but reasonably wide classes of BNs such as AND-OR BNs and BNs consisting of nested canalyzing functions, by combining recursive assignment techniques with SAT algorithms. However, these presented algorithms are not necessarily optimal. Therefore, improvement of the time complexity of these algorithms is left as an open problem. We have also seen that singleton attractor detection can be done in polynomial time for BNs with bounded treewidth and bounded maximum degree, using tree decomposition and dynamic programming as in many other polynomial-time algorithms for graphs with bounded treewidth. As for enumeration of singleton attractors, we have reviewed a simple recursive algorithm that works in $o(2^n)$ time in the average case.

Detection of a periodic attractor is much more difficult. We have seen that detection of an attractor with period two can be done in $o(2^n)$ time for AND-OR BNs. However, as far as we know, there is no $o(2^n)$ time algorithm that can detect an attractor with period three (or more) for reasonably wide classes of BNs. Therefore, development of such an algorithm is left as an open problem. Recently, in order to cope with this difficulty, an algorithm has been proposed which detects a long periodic attractor in $o(2^n)$ expected time, assuming that partial $0-1$ assignments in a desired attractor is probabilistically given in advance as *a priori* [15]. *This method succeeded in detecting attractors of large networks with long periods that were difficult to find by existing methods.*

From a practical viewpoint, many methods have been developed for detection/enumeration of singleton/periodic attractors for synchronous and asynchronous BNs. However, it is unclear which methods are the most useful in practice, where it may depend on structures of the target BNs. Therefore, rigorous comparison of state-of-the-art methods should be done. Furthermore, it seems that detection/enumeration of long attractors remains quite difficult also in practice. Therefore, practically efficient methods should be developed for detection/enumeration of long attractors.

CRediT authorship contribution statement

Tomoya Mori: Investigation, Writing – original draft. **Tatsuya Akutsu:** Conceptualization, Investigation, Writing – original draft.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Kauffman SA. Metabolic stability and epigenesis in randomly constructed genetic nets. J Theor Biol 1969;22(3):437–67. https://doi.org/10.1016/0022-5193(69)90015-0.
[2] Kauffman SA. Homeostasis and differentiation in random genetic control networks. Nature 1969;224(5215):177–8.
[3] Kauffman SA. The origin of order: self-organization and selection in evolution. New York: Oxford University Press; 1993.
[4] Kauffman SA. Metabolic stability and epigenesis in randomly constructed genetic nets. J Theoret Biol 2010;22:437–67. https://doi.org/10.1016/0022-5193(69)90015-0.
[5] Cheng D, Qi H, Li Z. Analysis and control of Boolean networks: a semi-tensor product approach. London, UK: Springer-Verlag; 2011.
[6] Akutsu T. Algorithms for analysis, inference, and control of Boolean networks. Singapore: World Scientific; 2018.
[7] Anthony M. Discrete mathematics of neural networks, selected topics. Philadelphia, PA, USA: SIAM; 2001.
[8] Münzner U, Klipp E, Krantz M. A comprehensive, mechanistically detailed, and executable model of the cell division cycle in Saccharomyces cerevisiae. Nat Commun 2019;10(1308). https://doi.org/10.1038/s41467-019-08903-w.
[9] Mori T, Flöttmann M, Krantz M, Akutsu T, Klipp E. Stochastic simulation of Boolean rxncon models: towards quantitative analysis of large signaling networks. BMC Syst Biol 2015;9(45). https://doi.org/10.1186/s12918-015-0193-8.
[10] Romers J, Thieme S, Münzner U, Krantz M. Using rxncon to develop rule-based models. Methods Mol Biol 2019;1945:71–118. https://doi.org/10.1007/978-1-4939-9102-0_4.
[11] Saadatpour A, Albert I, Albert R. Attractor analysis of asynchronous Boolean models of signal transduction networks. J Theor Biol 2010;266:641–56. https://doi.org/10.1016/j.jtbi.2010.07.022.
[12] Weinstein N, Mendoza L, Gitler I, Klapp J. A network model to explore the effect of the micro-environment on endothelial cell behavior during angiogenesis. Front Physiol 2017;8(960). https://doi.org/10.3389/fphys.2017.00960.
[13] Saadatpour A, Wang RS, Liao A, Liu X, Loughran TP, Albert I, et al. Dynamical and structural analysis of a T cell survival network identifies novel candidate therapeutic targets for large granular lymphocyte leukemia. PLoS Comput Biol 2011;7(11):1–15. https://doi.org/10.1371/journal.pcbi.1002267.
[14] About-Jaoudé W, Traynard P, Monteiro PT, Saez-Rodriguez J, Helikar T, Thieffry D, et al. Logical modeling and dynamical analysis of cellular networks. Front Genet 2016;7(94). https://doi.org/10.3389/fgene.2016.00094.
[15] Münzner U, Mori T, Krantz M, Klipp E, Akutsu T. Identification of periodic attractors in Boolean networks using a priori information. PLoS Comput Biol 2022;18(1):. https://doi.org/10.1371/journal.pcbi.1009702e1009702.
[16] Akutsu T, Kuhara S, Maruyama O, Miyano S. A system for identifying genetic networks from gene expression patterns produced by gene disruptions and overexpressions. Genome Inform 1998;9:151–60.
[17] Aracena J, Demongeot J, Goles E. Positive and negative circuits in discrete neural networks. IEEE Trans Neural Netw 2004;15(1):77–83. https://doi.org/10.1109/TNN.2003.821555.
[18] Akutsu T, Tamura T. On finding a fixed point in a Boolean network with maximum in degree 2. IEICE Trans Fundam 2009;92-A(8):1771–8. https://doi.org/10.1587/transfun.E92.A.1771.
[19] Melkman AA, Akutsu T. An improved satisfiability algorithm for nested analyzing functions and its application to determining a singleton attractor of a Boolean network. J Comput Biol 2013;20(12):958–69. https://doi.org/10.1089/cmb.2013.0060.
[20] Aracena J. On the number of fixed points in regulatory Boolean networks. Bull Math Biol 2008;70:1398–409. https://doi.org/10.1007/s11538-008-9304-7.
[21] Goles E, Salinas L. Sequential operator for filtering cycles in Boolean networks. Adv Appl Math 2010;45:346–58. https://doi.org/10.1016/j.aam.2010.03.002.
[22] Harris SE, Sawhill BK, Wuensche A, Kauffman S. A model of transcriptional regulatory networks based on biases in the observed regulation rules. Complexity 2002;7(4):23–40. https://doi.org/10.1002/cplx.10022.
[23] Jarrah AS, Raposa B, Laubenbacher R. Nested canalyzing, unate cascade, and polynomial functions. Physica D 2007;233(2):167–74. https://doi.org/10.1016/j.physd.2007.06.022.
[24] Kauffman SA, Peterson C, Samuelsson B, Troein C. Genetic networks with canalyzing Boolean rules are always stable. Proc Natl Acad Sci USA 2004;101 (49):17102–7. https://doi.org/10.1073/pnas.0407783101.
[25] Layne L, Dimitrova E, Macauley M. Nested canalyzing depth and network stability. Bull Math Biol 2004;74(2):422–33. https://doi.org/10.1007/s11538-011-9692-y.
[26] Li Y, Adeyeye JO, Murrugarra D, Aguilar B, Laubenbacher R. Boolean nested canalizing functions: a comprehensive analysis. Theor Comput Sci 2013;481:24–36. https://doi.org/10.1016/j.tcs.2013.02.020.
[27] Jarrah AS, Laubenbacher R, Veliz-Cuba A. The dynamics of conjunctive and disjunctive Boolean network models. Bull Math Biol 2010;72:1425–47. https://doi.org/10.1007/s11538-010-9501-z.
[28] Melliti T, Regnault D, Richard A, Sené S. On the convergence of Boolean automata networks without negative cycles. In: 19th International Workshop on Cellular Automata and Discrete Complex Systems (AUTOMATA 2013). p. 124–38. https://doi.org/10.1007/978-3-642-40867-0_9.
[29] Noual M, Regnault D, Sené S. Boolean networks synchronism sensitivity and XOR circulant networks convergence time. arXiv 2012;10.48550/arXiv. 1208.2767..
[30] Colón-Reyes O, Laubenbacher R, Pareigis B. Boolean monomial dynamical systems. Ann Comb 2005;8(4):425–39. https://doi.org/10.1007/s00026-004-0230-6.
[31] Park JW, Gao S. Monomial dynamical systems in # P-complete. Math J Interdiscip Sci 2012;1(1). 10.15415/mjis.2012.11008.
[32] Veliz-Cuba A, Laubenbacher R. The dynamics of semilattice networks. arXiv 2010;10.48550/arXiv. 1010.0359..
[33] Alcolei A, Perrot K, Sené S. On the flora of asynchronous locally non-monotonic Boolean automata networks. arXiv 2015;10.48550/arXiv. 1510.05452..
[34] Veliz-Cuba A, Aguilar B, Laubenbacher R. Dimension reduction of large sparse AND-NOT network models. Electron Notes Theor Comput Sci 2015;316 (16):83–95. https://doi.org/10.1016/j.entcs.2015.06.012.

[35] Aldana M, Coppersmith S, Kadanoff LP. Boolean dynamics with random couplings. In: Kaplan E, Marsden J, Sreenivasan K, editors. Perspectives and Problems in Nolinear Science. New York NY: Springer; 2003. p. 23–89. https://doi.org/10.1007/978-0-387-21789-5_2.

[36] Harvey I, Bossomaier T. Time out of joint: attractors in asynchronous random Boolean networks. In: Proceedings of 4th European Conference on Artificial Life. p. 67–75.

[37] Mochizuki A. An analytical study of the number of steady states in gene regulatory networks. J Theor Biol 2005;236(3):291–310. https://doi.org/10.1016/j.jtbi.2005.03.015.

[38] Drossel B, Mihaljev T, Greil F. Number and length of attractors in a critical Kauffman model with connectivity one. Phys Rev Lett 2005;94(8):. https://doi.org/10.1103/PhysRevLett.94.088701088701.

[39] Samuelsson B, Troein C. Superpolynomial growth in the number of attractors in Kauffman networks. Phys Rev Lett 2003;90(9):. https://doi.org/10.1103/PhysRevLett.90.098701098701.

[40] Thomas R. Regulatory networks seen as asynchronous automata: a logical description. J Theoretical Biol 1991;153(1):1–23. https://doi.org/10.1016/S0022-5193(05)80350-9.

[41] Mizera A, Pang J, Qu H, Yuan Q. Taming asynchrony for attractor detection in large Boolean networks. IEEE/ACM Trans Comput Biol Bioinform 2019;16(1):31–42. https://doi.org/10.1109/TCBB.2018.2850901.

[42] Giang TV, Akutsu T, Hiraishi K. An FVS-based approach to attractor detection in asynchronous random Boolean networks. IEEE/ACM Trans Comput Biol Bioinform 2022;19(2):806–18. https://doi.org/10.1109/TCBB.2020.3028862.

[43] Chatain T, Haar S, Paulevé L. Most permissive semantics of boolean networks. arXiv 2018;10.48550/arXiv. 1808.10240..

[44] de Jong H, Page M. Search for steady states of piecewise-linear differential equation models of genetic regulatory networks. IEEE/ACM Trans Comput Biol Bioinform 2008;5(2):208–22. https://doi.org/10.1109/TCBB.2007.70254.

[45] Dubrova E, Teslenko M. A SAT-based algorithm for finding attractors in synchronous Boolean networks. IEEE/ACM Trans Comput Biol Bioinform 2011;8(5):1393–9. https://doi.org/10.1109/TCBB.2010.20.

[46] Leone M, Pagnani A, Parisi G, Zagordi O. Finite size corrections to random Boolean networks. J Stat Mech 2006. https://doi.org/10.1088/1742-5468/2006/12/P12012.

[47] Balyo T, Heule M, Jarvisalo M. SAT competition 2016: recent developments. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence. p. 5061–3.

[48] Makino K, Tamaki S, Yamamoto M. Derandomizing the HSSW algorithm for 3-SAT. Algorithmica 2013;67(2):112–24. https://doi.org/10.1007/978-3-642-22685-4_1.

[49] Levy H, Low DW. A contraction algorithm for finding small cycle cutlets. J Algorithms 1988;9(4):470–93.

[50] Chen J, Fomin FV, Liu Y, Lu S, Villanger Y. Improved algorithms for feedback vertex set problems. J Comput Syst Sci 2008;74(7):1188–98. https://doi.org/10.1016/j.jcss.2008.05.002.

[51] Li J, Nederlof J. Detecting feedback vertex sets of size k in $o*(2.7^k)$ time. In: Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms (SODA). p. 971–89. https://doi.org/10.1137/1.9781611975994.58.

[52] Mochizuki A, Fiedler B, Kurosawa G, Saito D. Dynamics and control at feedback vertex sets. II: a faithful monitor to determine the diversity of molecular activities in regulatory networks. J Theor Biol 2013;335:130–46. https://doi.org/10.1016/j.jtbi.2013.06.009.

[53] Mori F, Mochizuki A. Expected number of fixed points in Boolean networks with arbitrary topology. Phys Rev Lett 2017;119(2). https://doi.org/10.1103/PhysRevLett.119.028301.

[54] Zhang SQ, Hayashida M, Akutsu T, Ching WK, Ng MK. Algorithms for finding small attractors in Boolean networks. EURASIP J Bioinform Syst Biol 2007;2007(1). https://doi.org/10.1155/2007/20180.

[55] Tamura T, Akutsu T. Detection a singleton attractor in a Boolean network utilizing SAT algorithms. IEICE Trans Fundamentals 2009;E92-A(2):493–501. https://doi.org/10.1587/transfun.E92.A.493.

[56] Yamamoto M. An improved $\tilde{O}(1.234^m)$-time deterministic algorithm for SAT. In: Proceedings of 16th International Symposium on Algorithms and Computation (Lecture Notes in Computer Science 3827). p. 644–53. https://doi.org/10.1007/11602613_65.

[57] Melkman AA, Tamura T, Akutsu T. Determining a singleton attractor of an AND/OR Boolean network in $O(1.587^n)$ time. Inf Process Lett 2010;110(14–15):565–9. https://doi.org/10.1016/j.ipl.2010.05.001.

[58] Chu H, Xiao M, Zhang Z. An improved upper bound for SAT. Theoret Comput Sci 2021;887:51–62. https://doi.org/10.1016/j.tcs.2021.06.045.

[59] Flum J, Grohe M. Parameterized complexity theory. Berlin: Springer; 2006.

[60] Fomin FV, Kratsch D. Exact exponential algorithms. Berlin: Springer; 2010.

[61] Akutsu T, Kosub S, Melkman AA, Tamura T. Finding a periodic attractor of a Boolean networks. IEEE/ACM Trans Comput Biol Bioinform 2012;9(5):1410–21. https://doi.org/10.1109/TCBB.2012.87.

[62] Chang CJ, Tamura T, Chao KM, Akutsu T. A fixed-parameter algorithm for detecting a singleton attractor in an AND/OR Boolean network with bounded treewidth. IEICE Trans Fundamentals 2015;98-A(1):384–90. https://doi.org/10.1587/transfun.E98.A.384.

[63] Freuder EC. Complexity of k-tree structured constraint satisfaction problems. In: Proceedings of the 8th AAAI Conference on Artificial Intelligence. p. 4–9.

[64] Just W. The steady state system problem is NP-hard even for monotone quadratic Boolean dynamical systems; 2006. Preprint available at http://www.ohio.edu/people/just/publ.html..

[65] Devloo V, Hansen P, Labbé M. Identification of all steady states in large networks by logical analysis. Bull Math Biol 2003;65(6):1025–51. https://doi.org/10.1016/S0092-8240(03)00061-2.

[66] Inoue K. Logic programming for Boolean networks. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence. p. 924–30. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-160.

[67] Abdallah EB, Folschette M, Roux O, Magnin M. Asp-based method for the enumeration of attractors in non-deterministic synchronous and asynchronous multi-valued networks. Algorithms Mol Biol 2017;12(20). https://doi.org/10.1186/s13015-017-0111-2.

[68] Dubrova E, Teslenko M, Martinelli A. Kaufmann networks: analysis and applications. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design. p. 479–84. https://doi.org/10.1109/ICCAD.2005.1560115.

[69] Garg A, Cara AD, Xenarios I, Mendoza L, Micheli GD. Synchronous versus asynchronous modeling of gene regulatory networks. Bioinformatics 2008;24(17):1917–25. https://doi.org/10.1093/bioinformatics/btn336.

[70] Zheng D, Yang G, Li X, Wang Z, Liu F, He L. An efficient algorithm for computing attractors of synchronous and asynchronous Boolean networks. PLoS ONE 2013;8(4):e60593.

[71] Akutsu T, Zhao Y, Hayashida M, Tamura T. Integer programming-based approach to attractor detection and control of Boolean networks. IEICE Trans Inf Syst 2012;E95-D(12):2960–70. https://doi.org/10.1587/transinf.E95.D.2960.

[72] Klarner H, Bockmayr A, Siebert H. Computing symbolic steady states of Boolean networks. In: Cellular Automata. ACRI 2014. Lecture Notes in Computer Science; vol. 8751. Springer, Cham; 2014, p. 561–570. DOI: 10.1007/978-3-319-11520-7_59..

[73] Kobayashi K, Hiraishi K. ILP/SMT-based method for design of Boolean networks based on singleton attractors. IEEE/ACM Trans Comput Biol Bioinform 2010;11(4):1253–9. https://doi.org/10.1109/TCBB.2014.2325011.

[74] Veliz-Cuba A. Reduction of Boolean network models. J Theor Biol 2011;289:167–72. https://doi.org/10.1016/j.jtbi.2011.08.042.

[75] Veliz-Cuba A, Aguilar B, Hinkelmann F, Laubenbacher R. Steady state analysis of Boolean molecular network models via model reduction and computational algebra. BMC Bioinformatics 2014;15(221).

[76] He Q, Xia Z, Lin B. An efficient approach of attractor calculation for large-scale Boolean gene regulatory networks. J Theor Biol 2016;408:137–44. https://doi.org/10.1016/j.jtbi.2016.08.006.

[77] Saadatpour A, Albert R, Reluga TC. A reduction method for Boolean network models proven to conserve attractors. SIAM J Appl Dyn Syst 2013;12(4):1997–2011. https://doi.org/10.1137/13090537X.

[78] Beneš N, Brim L, Pastva S, Šafránek D. Computing bottom SCCs symbolically using transtion guided reduction. In: International Conference on Computer Aided Verification. Cham: Springer; 2021. p. 505–28. https://doi.org/10.1007/978-3-030-81685-8_24.

[79] Gao Z, Chen X, Başar T. Stability structures of conjunctive Boolean networks. Automatica 2018;89:8–20. https://doi.org/10.1016/j.automatica.2017.11.017.

[80] Chen X, Gao Z, Başar T. Asymptotic behavior of conjunctive Boolean networks over weakly connected digraphs. IEEE Trans Automat Contr 2019;65(6):2536–49. https://doi.org/10.1109/TAC.2019.2930675.

[81] Irons DJ. Improving the efficiency of attractor cycle identification in Boolean networks. Physica D 2006;217(1):7–21. https://doi.org/10.1016/j.physd.2006.03.006.

[82] Su C, Pang J, Paul S. Towards optimal decomposition of Boolean networks. IEEE/ACM Trans Comput Biol Bioinform 2021;18(6):2167–76. https://doi.org/10.1109/TCBB.2019.2914051.

[83] Zañudo JGT, Albert R. An effective network reduction approach to find the dynamical repertoire of discrete dynamic networks. Chaos 2013;23(025111). https://doi.org/10.1063/1.4809777.

[84] Klarner H, Siebert H. Approximating attractors of Boolean networks by iterative CTL model checking. Front Bioeng Biotechnol 2015;3(130). https://doi.org/10.3389/fbioe.2015.00130.

[85] Choo SM, Cho KH. An efficient algorithm for identifying primary phenotype attractors of a large-scale Boolean network. BMC Syst Biol 2016;10(95). https://doi.org/10.1186/s12918-016-0338-4.

[86] Tamaki H. A directed path-decomposition approach to exactly identifying attractors of Boolean networks. In: Proceedings of 10th International Symposium on Communication and Information Technologies. p. 844–9. https://doi.org/10.1109/ISCIT.2010.5665106.

[87] Skodawessely T, Klemm K. Finding attractors in asynchronous Boolean dynamics. Adv Complex Syst 2011;14:439–49. https://doi.org/10.1142/S0219525911003098.

[88] Lu J, Li H, Liu Y, Li F. Survey on semi-tensor product method with its applications in logical networks and other finite-valued systems. IET Control Theory Appl 2017;11(13):2040–7. https://doi.org/10.1049/iet-cta.2016.1659.

[89] Zhao Y, Ghosh BK, Cheng D. Control of large-scale Boolean networks via network aggregation. IEEE Trans Neural Netw Learn Syst 2016;27(7):1527–36. https://doi.org/10.1109/TNNLS.2015.2442593.

[90] Liu X, Wang Y, Shi N, Ji Z, He S. GAPORE: Boolean network inference using a genetic algorithm with novel polynomial representation and encoding scheme. Knowl-Based Syst 2021;228:. https://doi.org/10.1016/j.knosys.2021.107277107277.