PAPER

A Fixed-Parameter Algorithm for Detecting a Singleton Attractor in an AND/OR Boolean Network with Bounded Treewidth

Chia-Jung CHANG[†], *Nonmember*, Takeyuki TAMURA^{††a)}, *Member*, Kun-Mao CHAO^{†,†††}, *Nonmember*, and Tatsuya AKUTSU^{††}, *Member*

SUMMARY The Boolean network can be used as a mathematical model for gene regulatory networks. An attractor, which is a state of a Boolean network repeating itself periodically, can represent a stable stage of a gene regulatory network. It is known that the problem of finding an attractor of the shortest period is NP-hard. In this article, we give a fixed-parameter algorithm for detecting a singleton attractor (SA) for a Boolean network that has only AND and OR Boolean functions of literals and has bounded treewidth k . The algorithm is further extended to detect an SA for a constant-depth nested canalizing Boolean network with bounded treewidth. We also prove the fixed-parameter intractability of the detection of an SA for a general Boolean network with bounded treewidth.

key words: Boolean network, attractor, partial k -tree, fixed-parameter algorithm

1. Introduction

The *Boolean network* (BN) is known as a discrete mathematical model of gene regulatory networks [8]. In a BN, each vertex corresponds to a gene and takes one of two values 0 and 1, where 0 (resp., 1) means that the corresponding gene is inactive (resp., active). The value of a vertex at a given time step is determined according to a regulation rule, which is a Boolean function of the values of the predecessors of the vertex at the previous time instant. The values of vertices are updated synchronously, and the (global) *state* of a network at a given time step is the vector of its vertex values. Beginning from any initial state, the system eventually falls into an *attractor*, which is classified into two types: a *singleton attractor* corresponding to a stable state, and a *periodic attractor* corresponding to a sequence of states that repeats periodically.

The analysis of attractors is an important research topic in bioinformatics and computational biology since Kauffman pointed out the correspondence between different attractors and different types of cells [8]. Therefore many heuristic algorithms have been developed to detect and/or enumerate attractors [3], [4], [6], [11].

It is known that the problem of finding an attractor of the shortest period is NP-hard even for BNs with maximum in-degree 2 consisting of AND/OR of literals [12]. Due to this hardness and the fact that there exist 2^n global states for a BN with n vertices, previous theoretical studies focused on the development of $o(2^n)$ -time algorithms.

Since it is quite hard to develop such algorithms for general BNs, some restrictions were assumed on types of Boolean functions in all studies. For example, an $O(1.587^n)$ -time algorithm and an $O(1.985^n)$ -time algorithm were developed for detection of a singleton attractor [9] and an attractor of period 2 [1], respectively, both for *AND/OR BNs* which are BNs consisting of Boolean functions restricted to conjunctions and disjunctions of literals.

An $O(n^{2p(k+1)}poly(n))$ -time algorithm was also developed for finding an attractor of period p of a BN having bounded treewidth k and consisting of nested canalizing functions [8], where p and k are constants, and nested canalizing functions are a super class of AND/OR functions [1]. They also presented a fixed-parameter algorithm (precisely, an algorithm working in $O(g(p, k, d)poly(n))$ time where $g(p, k, d)$ depends only on p, k, d for a general BN with bounded degree d and bounded treewidth [1]. However, it is unknown whether there exists a fixed-parameter algorithm even for an AND/OR BN with bounded treewidth but without degree constraint. In this paper, we first present a fixed-parameter algorithm for detection of a singleton attractor in an AND/OR BN with bounded treewidth. We further extend the algorithm for detection of a singleton attractor in a constant-depth nested canalizing function BN with bounded treewidth. Finally, we prove that detection of singleton attractor for a general BN with bounded treewidth is W[1] hard for parameter k (i.e., fixed-parameter intractable for parameter k).

2. Preliminaries

A BN $N(V, F)$ consists of a set V of n vertices and a corresponding set $F = \{f_v : v \in V\}$ of n Boolean functions. Let $v(x) \in \{0, 1\}$ represent the value of a vertex v at time x , and denote by $\mathbf{v}(x) = \langle v(x) : v \in V \rangle$ the *state of the network* at time x . The values of all vertices are updated simultaneously according to the corresponding Boolean functions, $v(x+1) = f_v(\mathbf{v}(x))$. Figure 1(a) gives an example. A directed graph $G(V, E)$ can be associated with the network, with a directed edge $(u, v) \in E$ if and only if f_v depends on u ,

Manuscript received May 15, 2014.

Manuscript revised August 21, 2014.

[†]The authors are with the Department of Computer Science and Information Engineering, National Taiwan University, No.1, Sec.4, Roosevelt Road, 10617 Taipei, Taiwan (R.O.C.).

^{††}The authors are with the Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji-shi, 611-0011 Japan.

^{†††}The author is with the Graduate Institute of Biomedical Electronics and Bioinformatics, National Taiwan University, No.1, Sec.4, Roosevelt Road, 10617 Taipei, Taiwan (R.O.C.).

a) E-mail: tamura@kuicr.kyoto-u.ac.jp

DOI: 10.1587/transfun.E98.A.384

Fig. 1 (a) A BN. (b) The directed graph associated with (a). (c) A tree decomposition of (b) and the consistent assignments for the nodes where $0/1_\omega$ represents $\phi_t(v) = \omega$ and $\hat{\phi}_t(v) = 0/1$. $\phi_{t_A} = (0, 0, 0)$ is a candidate assignment but not a consistent assignment because there is no consistent assignment of t_C that is compatible with it. (d) SAs of (a).

where edges may be self-loops. Figure 1(b) gives an example. The initial assignment of values $\mathbf{v}(1) = \langle v(1) : v \in V \rangle$ uniquely determines the state of the network at all $x > 1$. An initial state is called a *periodic attractor with period p* if $\mathbf{v}(1) = \mathbf{v}(p + 1)$ and $\mathbf{v}(1) \neq \mathbf{v}(q)$ holds for all $1 < q < p + 1$. An attractor with period $p = 1$ is called a *singleton attractor*, SA in short. Figure 1(d) shows SAs for the BN in Fig. 1(a).

Problem 1: Detection of SA

Given a BN, decide whether or not there exists a singleton attractor and output one if it exists.

A *tree decomposition* of a graph $G(V, E)$ is a pair $\langle T(V_T, E_T), (B_t)_{t \in V_T} \rangle$, where $T(V_T, E_T)$ is a rooted tree and $(B_t)_{t \in V_T}$ is a family of subsets of V such that

- for every $v \in V$, $B^{-1}(v) = \{t \in V_T | v \in B_t\}$ is nonempty and connected in T , and
- for every edge $\{u, v\} \in E$, there exists $t \in V_T$ such that $u, v \in B_t$.

The *width* of the decomposition is defined as $\max_{t \in V_T} (|B_t| - 1)$ and the *treewidth* of G is the minimum of the widths among all the tree decompositions of G . Graphs with treewidth at most k are also known as *partial k-trees* [5]. Given a partial k -tree where k is a constant, a width k tree decomposition can be constructed in linear time, i.e., $O(f(k)n)$ [13]. We assume that the tree decomposition has been constructed before we apply our algorithms. The tree in Fig. 1(c) is a partial 2-tree of the graph in Fig. 1(b).

A *nested canalizing function*, *NC-function* for short, can be represented as

$$f_v = l_1 \vee \dots \vee l_{k_1-1} \vee (l_{k_1} \wedge \dots \wedge l_{k_2-1} \wedge (l_{k_2} \vee \dots \vee l_{k_3-1} \vee \dots))$$

where l_i is the literal of the i -th predecessor of v and $1 \leq k_1 < k_2 < \dots < k_c$ [7]. We call a BN consisting of only NC-functions an *NC-BN*. A constant-depth NC-BN is an NC-BN where the depth (i.e. k_c) of each NC-function is no more

than a constant c .

3. The Algorithm for Detecting a Singleton Attractor in an AND/OR Boolean Network with Bounded Treewidth

Let $\langle T(V_T, E_T), (B_t)_{t \in V_T} \rangle$ be a tree decomposition of $G(V, E)$ associated to a given BN $N(V, F)$. Let $des(t)$ denote the set of descendants of $t \in V_T$ including t . Let $V_t = \bigcup_{t' \in des(t)} B_{t'}$. For each $t \in V_T$, $p(t)$ denotes the parent node of t in V_T .

Instead of enumerating all possible 0-1 assignments to V to find singleton attractors, we could restrict the enumeration in each B_t . We will show that there is a process to examine a partial 0-1 assignment to B_t to see if it could be a part of a singleton attractor. For each $t \in V_T$, ϕ_t denotes a function from B_t to $\{0, 1, \omega\}$, where we call such ϕ_t an *assignment*. The symbol ω is a mark to represent the status of a vertex in B_t during the process and each assignment ϕ_t actually corresponds to a 0-1 assignment $\hat{\phi}_t$ where

$$\hat{\phi}_t(v) = \begin{cases} \phi_t(v), & \text{if } \phi_t(v) \in \{0, 1\}, \\ 0, & \text{if } \phi_t(v) = \omega \text{ and } f_v \text{ is AND,} \\ 1, & \text{if } \phi_t(v) = \omega \text{ and } f_v \text{ is OR.} \end{cases}$$

We also use $0/1_\omega$ to represent that $\phi_t(v) = \omega$ and $\hat{\phi}_t(v) = 0/1$.

For a 0-1 assignment α to $V' \subseteq V$, $\phi_t(v)$ is called *validated* (by α on V') if $\phi_t(v) \in \{0, 1\}$ and the value is uniquely determined as $\alpha(v)$ from α regardless of an assignment to $V - V'$. In our algorithm, V' could be B_t or $V_t \cup B_t$ and α could be $\hat{\phi}_t$ or the combination of assignments on t and $des(t)$. For example, in Fig. 1(c), assuming that α to (a, e) equals $\phi_{t_D} = (0, 1)$, $\phi_{t_D}(a)$ and $\phi_{t_D}(e)$ are both validated by α because $f_a(\alpha) = 0 = \alpha(a)$ and $f_e(\alpha) = 1 = \alpha(e)$. The symbol ω represents that the corresponding vertex is not validated in the current assignment. The purpose of our method is to find an assignment to V that validates all possible $\phi_t(v)$ s by combining partial assignments that validate some part of the vertices respectively.

We say that α *violates* a Boolean function f_v (assigned to vertex v) if the value (b_v) of f_v is uniquely determined by α but $b_v \neq \alpha(v)$. α cannot be part of an SA if it violates any Boolean functions of the vertices. For example, in Fig. 1(c), assuming that α to (a, e) equals $\phi_{t_D} = (1, 1)$, α violates f_e since $f_e(\alpha) = 0 \neq \alpha(e)$.

It is to be noted that we need not validate if $\hat{\phi}_t(v) = 1$ and v is an AND vertex, or, $\hat{\phi}_t(v) = 0$ and v is an OR vertex. Such $\hat{\phi}_t(v)$ is treated as validated because Boolean values of its input vertices are uniquely determined (in a singleton attractor). Therefore, $\phi_t(v) = \omega$ holds only if v is an AND vertex and $\hat{\phi}_t(v) = 0$, or v is an OR vertex and $\hat{\phi}_t(v) = 1$. It is enough for such a vertex to examine whether α does not violate f_v .

ϕ_t is called *consistent* if the following is satisfied

- (1) $\phi_t(v) \in \{0, 1\}$ holds for all $v \in B_t - B_{p(t)}$, where we let $B_{p(t)} = \emptyset$ if t is the root, and

[†]We use nodes and vertices for $\langle T(V_T, E_T), (B_t)_{t \in V_T} \rangle$ and $N(V, F)$, respectively.

- (2) there exists a 0-1 assignment α to V_t such that $\hat{\phi}_t(v) = \alpha(v)$ holds for all $v \in B_t$, α does not violate any Boolean function assigned to V_t , and $\phi_t(v)$ is validated for all v such that $\phi_t(v) \in \{0, 1\}$ by α on V_t .

For a consistent assignment, 0/1 means $\phi_t(v)$ is validated and ω means $\phi_t(v)$ is not validated nor violated by the assignment α in (2). For example, in Fig. 1(c), assuming $\alpha = \phi_{t_C} = (0_\omega, 1, 1)$, α does not validate nor violate $\phi_{t_C}(a)$. Let A_t be the set of consistent assignments to B_t . There exists a singleton attractor iff $A_r \neq \emptyset$ for the root r of T . Furthermore, such a singleton attractor can be retrieved by using the standard traceback technique. We describe below how to compute A_t by dynamic programming from leaves to the root in V_T . For each leaf t , A_t is determined by $A_t = \{\phi_t \mid \phi_t \text{ satisfies conditions (1) and (2) for } \alpha = \hat{\phi}_t\}$.

Let $t_1, \dots, t_d$ be the children of t . We call ϕ_t and ϕ_{t_i} are *compatible* if, for each $v \in B_t \cap B_{t_i}$, $\phi_t(v) = \phi_{t_i}(v)$ holds, or $\phi_t(v) \in \{0, 1\}$, $\phi_{t_i}(v) = \omega$ and $\hat{\phi}_t(v) = \hat{\phi}_{t_i}(v)$ hold. For example, in Fig. 1(c), assuming $\phi_{t_A} = (0, 1, 0)$ and $\phi_{t_C} = (0_\omega, 1, 1)$, ϕ_{t_A} and ϕ_{t_C} are compatible. We say that ϕ_t is a *candidate* assignment if $\phi_t(v) \in \{0, 1\}$ holds for any vertex in $B_t - B_{p(t)}$, ϕ_t does not violate any function assigned to B_t , and $\phi_t(v)$ is validated by ϕ_t for any vertex $B_t - B_{p(t)} - \cup_{j=1, \dots, d} B_{t_j}$. For an OR vertex $v \in B_t$, $\phi_t(v) = 1$ is validated by ϕ_t if there exists some input vertex $u \in B_t$ such that f_v contains the positive literal of u and $\hat{\phi}_t(u) = 1$, or f_v contains the negative literal of u and $\hat{\phi}_t(u) = 0$. The validation is similar for an AND vertex. For each candidate assignment, we maintain a 0-1 table $X(x_1, \dots, x_h)$ where $\{v_{j_1}, \dots, v_{j_h}\} = B_t$ (i.e., X is a 0-1 table having $2^{|B_t|}$ entries). Each entry is a state of validation where $x_k = 1$ means $\phi_t(v_{j_k})$ has been validated or $\phi_t(v_{j_k}) = \omega$ and $x_k = 0$ means otherwise. An entry $X(x_1, \dots, x_h) = 1$ means the state can be achieved by the combination of ϕ_t and consistent assignments of different children of t . It is to be noted that since the degree of $t \in V_T$ may be $O(n)$, the computation time would be exponential of n if the table X is not used and all combinations of ϕ_{t_i} are considered for $1 \leq i \leq O(n)$. The following is a pseudocode to find A_r (see also Fig. 2).

Fig. 2 The validation of an assignment. Part A ($B_t \cap B_{p(t)}$) can be validated later, Part B ($B_t - B_{p(t)} - \cup_j B_{t_j}$) must be validated by ϕ_t , and Part C ($(B_t \cap (\cup_j B_{t_j})) - B_{p(t)}$) must be validated by $\phi_t, \phi_{t_1}, \phi_{t_2}$.

Procedure *FpAttAND/OR*

for all leaves in V_T **do** compute A_t ;

for all internal nodes t in V_T **do**

$A_t \leftarrow \emptyset$;

for all candidate assignments ϕ_t **do**

for all $(x_1, \dots, x_h) \in \{0, 1\}^h$ **do**

$X(x_1, \dots, x_h) \leftarrow 0$;

$X(x_1, \dots, x_h) \leftarrow 1$ where $x_g = 1$ iff $\phi_t(v_{j_g})$ has already been validated, or $\phi_t(v_{j_g}) = \omega$;

for $i = 1$ **to** d **do**

$Y \leftarrow X$;

if there is no $\phi_{t_i} \in A_{t_i}$ compatible with ϕ_t

then break and examine next ϕ_t ;

for all $\phi_{t_i} \in A_{t_i}$ compatible with ϕ_t **do**

let $z_g = 1$ if $\phi_{t_i}(v_{j_g}) \in \{0, 1\}$,

otherwise $z_g = 0$;

for all $(x_1, \dots, x_h)$ with $X(x_1, \dots, x_h) = 1$ **do**

$Y(\max(x_1, z_1), \dots, \max(x_h, z_h)) \leftarrow 1$;

$X \leftarrow Y$;

if $X(1, 1, \dots, 1) = 1$ **then** $A_t \leftarrow A_t \cup \{\phi_t\}$

Theorem 1: Given its tree decomposition, the singleton attractor detection problem for an AND/OR BN with bounded treewidth k can be solved in $O(18^k \text{poly}(n))$ time.

Proof. It is to be noted that in each iteration of t , for an OR vertex $v \in B_t$, $\phi_t(v) = 1$ is validated iff $\phi_t(v)$ is validated by ϕ_t or, $\phi_{t_i}(v) = 1$ for some $\phi_{t_i} \in A_{t_i}$ for some i . The former condition can be checked at the part of $x_g = 1$, and the latter condition can be checked at the part of $z_g = 1$ in the procedure, where $z_g = 1$ means that v_{j_g} is newly validated (unless it has already been validated).

To prove the correctness of *FpAttAND/OR*, it is sufficient to prove that A_t contains all consistent assignments of t . A consistent assignment is a candidate assignment since a candidate assignment satisfies condition (1) of a consistent assignment, which is “ $\phi_t(v) \in \{0, 1\}$ holds for all $v \in B_t - B_{p(t)}$.” For a candidate assignment ϕ_t , the vertices in Part B (see Fig. 2) can be validated by ϕ_t itself. ϕ_t can be added to A_t when there is a consistent assignment ϕ_{t_i} for each child t_i of t that is compatible with ϕ_t and all $\phi_t(v_{j_g}) \in \{0, 1\}$ is validated by at least one of ϕ_{t_i} . Since $\phi_t(v) \in \{0, 1\}$ holds for all $v \in B_t - B_{p(t)}$, the vertices in Part C (see Fig. 2) have been validated. A portion of the vertices in Part A (see Fig. 2) might also have been validated in the same step. For the others ($\phi_t(v) = \omega$) in Part A, neither ϕ_t nor any ϕ_{t_i} violates their Boolean functions because ϕ_t is a candidate assignment and any ϕ_{t_i} is a consistent assignment. These vertices can be validated later. For each ϕ_{t_i} , since ϕ_{t_i} is a consistent assignment, there is a 0-1 assignment α_{t_i} that satisfies condition (2) of a consistent assignment. We can combine $\hat{\phi}_t$ and all α_{t_i} to form a 0-1 assignment α_t since ϕ_t is compatible with all ϕ_{t_i} and $V_{t_i} \cap B_t = B_{t_i} \cap B_t$ for all t_i in a tree decomposition. It is trivial that α_t satisfies the conditions of a consistent assignment.

Since the number of possible ϕ_t is bounded by 3^{k+1} per t and the size of table X is bounded by 2^{k+1} for partial k -trees, the computation time of *FpAttAND/OR* can be bounded by

$$O(3^k(2^k + n3^k2^k))poly(n) = O(18^k poly(n)). \quad \square$$

4. A Fixed-Parameter Algorithm for Detecting a Singleton Attractor in a Constant-Depth NC-BN with Bounded Treewidth

In this section, we extend the *FpAttAND/OR* procedure so that the SA detection problem for a constant-depth NC-BN with bounded treewidth k can be solved in $O(f(ck)poly(cn))$ time, where c is the maximum number of layers of the nested canalyzing functions.

The input of the problem is a BN $N(V, F)$, which consists of a set V of n vertices and a corresponding set $F = \{f_v : v \in V\}$ of n c -depth nested canalyzing functions. Let $\langle T(V_T, E_T), (B_i)_{i \in V_T} \rangle$ be one of its tree decomposition.

The algorithm is quite simple. First, the constant-depth NC-BN N is transformed to an AND/OR BN $N'(V', F')$ by introducing vertices that represent the parentheses in the nested canalyzing functions (we call them parenthesis vertices). For example, if $f_v = v_1 \vee v_2 \vee (v_3 \wedge v_4 \wedge v_5 \wedge (v_6 \vee (v_7 \wedge v_8)))$, three vertices v^1, v^2 and v^3 are added and $f'_v = v_7 \wedge v_8$, $f'_{v^2} = v_6 \vee v^3$, $f'_{v^1} = v_3 \wedge v_4 \wedge v_5 \wedge v^2$ and $f'_v = v_1 \vee v_2 \vee v^1$. The idea is illustrated in Fig. 3. N' has less than or equal to cn vertices and can be decomposed as a partial $(ck + c - 1)$ -tree $\langle T(V_T, E_T), (B'_i)_{i \in V_T} \rangle$. The tree decomposition of N' can be achieved by adding the parenthesis vertices in the same B'_i 's as their corresponding vertices. In the above example, if $v \in B_i$, $\{v, v^1, v^2, v^3\} \subset B'_i$.

Theorem 2: The singleton attractor detection problem for a constant- c -depth NC-BN with bounded treewidth k can be solved in $O(f(ck)poly(cn))$ time, where $f(ck)$ is a function depending only on c and k .

Proof. If there is a singleton attractor in N , there is a 0-1 assignment α to V such that $\alpha(v) = f_v(\alpha)$. Assignment α can be extended to α' on V' by assigning the parenthesis vertices $v = f'_v(\alpha')$ in a bottom-up order. Assignment α' is a singleton attractor of N' .

If there is a singleton attractor α' in N' , let $\alpha = \alpha'(V)$, where $\alpha'(V)$ is a function from $\{0, 1\}^{|V'|}$ to $\{0, 1\}^{|V|}$. Assignment α will be a singleton attractor of N according to the definition of N .

Fig. 3 Transforming a nested canalyzing function to multiple AND/OR Boolean functions. The grey vertices are the input of the white vertices. The vertices in the same grey scales are in the same layers of the nested canalyzing function. Dashed border vertices are added to represent the parentheses.

By utilizing the *FpFindAtt* procedure for N' , the problem can be solved in $O(f(ck)poly(cn))$ time. $\square$

5. Fixed-parameter Intractability of the Detection of Singleton Attractor of a BN with Bounded Treewidth

In this section, we are going to prove the fixed-parameter intractability of detection of singleton attractor of a general Boolean network with bounded treewidth k . The proof is by reducing the k -clique problem to this problem in polynomial time.

Problem 2: The k -clique problem

Given an undirected graph $G_c(V_c, E_c)$ and a number k , decide whether or not there exists a clique of size k in G_c and output one if it exists.

The idea of the reduction is to build a BN $N(V, F)$ with bounded treewidth given $G_c(V_c, E_c)$ and k . If there exists an SA in N , there should exist a clique of size k in G_c , and vice versa. For simplification, we use v to represent $v(t)$ in the following Boolean functions.

The construction of $N(V, F)$ given G_c is described as follows

Given $G_c(V_c, E_c)$ and k ,

- $V = \{R, X_1, X_2, \dots, X_k\} \cup \{q_i, q'_i | q \in V_c, 1 \leq i \leq k\}$,
- $f_R = \bar{R} \vee (X_1 \wedge X_2 \dots \wedge X_k)$,
- $f_{X_i} = \bigvee_{q \in V_c} (q_i \wedge (\bigwedge_{p \in V_c, (p,q) \notin E_c} \bar{p}_i \wedge \bar{p}'_i))$ for each $1 \leq i \leq k$,
- $f_{q'_i} = q'_i \wedge \bar{q}_i \wedge (\bigvee_{j \neq i} q_j)$ for each $q \in V_c$ and $1 \leq i \leq k$,
- $f_{q_i} = (q_i \wedge (\bigwedge_{j \neq i} \bar{q}_j)) \vee (\bar{q}'_i \wedge \bar{q}_i \wedge (\bigvee_{j \neq i} q_j \vee q'_j))$ for each $q \in V_c$ and $1 \leq i \leq k$.

For example, given a G_c , which consists of four vertices $\{a, b, c, d\}$ and four edges $\{(a, b), (b, c), (a, c), (a, d)\}$, and a $k = 3$,

- $f_R = \bar{R} \vee (X_1 \wedge X_2 \wedge X_3)$
- $f_{X_1} = a_1 \vee (b_1 \wedge \bar{d}_1 \wedge \bar{d}'_1) \vee (c_1 \wedge \bar{d}_1 \wedge \bar{d}'_1) \vee (d_1 \wedge \bar{b}_1 \wedge \bar{b}'_1 \wedge \bar{c}_1 \wedge \bar{c}'_1)$,
- $f_{a'_1} = a'_1 \wedge \bar{a}_1 \wedge (a_2 \vee a_3)$,
- $f_{a_1} = (a_1 \wedge \bar{a}_2 \wedge \bar{a}_3) \vee (\bar{a}'_1 \wedge \bar{a}_1 \wedge (a_2 \vee a'_2 \vee a_3 \vee a'_3))$

The remaining Boolean functions (e.g. $f_{X_2}, f_{a_2}, f_{b_3}$) are built in the same way. Figure 4 gives an example of the constructed Boolean network. It is a partial $3k - 1$ -tree since it can be decomposed as the tree in Fig. 5. The construction time grows polynomially with the size of G_c and the number k .

Lemma 3: If there exists a clique of size k in G_c , there exists a corresponding singleton attractor in N .

Proof. Say that the induced subgraph of $Q \subseteq V_c$ is a clique of size k and the vertices of Q are ranked arbitrarily. For $1 \leq i \leq k$, if q is the i -th vertex of Q , we set $q_i = 1$ and $q'_j = 1$ for any other $j \neq i$ (e.g. $a_1 = a'_2 = a'_3 = b'_1 = b_2 = b'_3 = c'_1 = c'_2 = c_3 = 1$). We also set $R = X_1 = \dots = X_k = 1$

Fig. 4 An illustration of the constructed Boolean network. For every vertex $q \in V_c$, its corresponding vertices $q_i, q'_i, 1 \leq i \leq k$ are connected with each other. We only draw such edges of one vertex for simplification. The direction of the edges and self-loop edges are not illustrated since they do not affect the result of the tree decomposition.

Fig. 5 The tree decomposition of the Boolean network in Fig. 4. It is a partial $3k - 1$ -tree.

and set the values of all the other vertices 0.

According to the assignment, for values of $q'_i q_i$, 01 means that the vertex q is the i -th vertex of the clique Q , 10 means that q belongs to the clique but it is not the i -th vertex, 00 means that q does not belong to the clique, and there is no 11.

$R(t+1) = 1$ because the clause $(X_1 \wedge X_2 \dots \wedge X_k) = 1$. For $X_i(t+1)$, there must exist a vertex $q \in V_c$ that is the i -th vertex of the clique and $q_i = 1$ according to the above assignment. All the vertices in V_c that are not connected to q should not appear in the clique, so the clause $(\bigwedge_{p \in V_c, (p,q) \notin E_c} \bar{p}_i \wedge \bar{p}'_i)$ should be 1 because the values of $p'_i p_i$ are all 00. Therefore, $X_i(t+1) = 1$.

The values of $q'_i(t+1)$ and $q_i(t+1)$ are discussed together. If the values of $q'_i q_i$ are 01, the values of all the other $q'_j q_j$ s, $j \neq i$ are 10. Therefore, the clause $(\bigwedge_{j \neq i} \bar{q}_j)$ should be 1 and thus the value of $q_i(t+1)$ is 1. The value of $q'_i(t+1)$ should be 0 because $q'_i = 0$ and the $\wedge$ operation. If the values of $q'_i q_i$ are 10, there should exist one pair of vertices $q'_j q_j = 01$, which makes $q'_i(t+1) = 1$, and $q_i(t+1)$ is always 0 given such $q'_i q_i$. If the values of $q'_i q_i$ are 00, the values of all the other $q'_j q_j$ s are 00, which make the values of $q'_i(t+1)q_i(t+1)$ become 00.

This assignment satisfies the condition of the singleton attractor $\mathbf{v}(t+1) = \mathbf{v}(t)$. $\square$

Now we are going to prove the other direction of the reduction. If there exists a singleton attractor for the constructed BN, we can find a clique of size k in G_c . Instead of giving a direct proof, we first prove some lemmas to support it.

Lemma 4: If there exists a singleton attractor for N , in its initial state, $X_i = 1$ for each $1 \leq i \leq k$.

Proof. $R(t+1) = 1$ when $R = 0$. Therefore, $R = 1$ if there exists an SA. To make $R(t+1) = 1$ when $R = 1$, the value of the clause $(X_1 \wedge X_2 \dots \wedge X_k)$ should be 1. $\square$

Lemma 5: If there exists a singleton attractor for N , in its initial state, the values of its $q'_i q_i$ cannot be 11 for any $q \in V_c$ and $1 \leq i \leq k$.

Proof. The Boolean function $f_{q'_i}$ proves this lemma. $\square$

Lemma 6: If there exists a singleton attractor for N , in its initial state, whenever the values of one $q'_i q_i$ are 00, the values of all the other $q'_j q_j$ are also 00.

Proof. By the definition of $f_{q'_i}$, $q'_i(t+1) = 0$ whenever $q'_i = 0$. To make $q_i(t+1) = 0$ when $q'_i q_i = 00$, the clause $(\bigvee_{j \neq i} q_j \vee q'_j)$ of $f_{q'_i}$ should be 0, which completes the proof. $\square$

Lemma 7: If there exists a singleton attractor for N , in its initial state, whenever the values of one $q'_i q_i$ are 01, the values of $q'_j q_j$ s for all $j \neq i$ should be 10.

Proof. According to the definition of $f_{q'_i}$, $q'_i(t+1) = 0$ whenever $q'_i = 0$. To make $q_i(t+1) = 1$, the clause $(\bigwedge_{j \neq i} \bar{q}_j)$ should be 1 and the values of all q_j ($j \neq i$) should be 0. According to Lemma 6, if there is one pair of $q'_j q_j = 00$, all the other such pairs should be 00, which conflicts the fact that $q'_i q_i = 01$. Therefore, the values of all the pairs $q'_j q_j$ s, $j \neq i$, should be 10. $\square$

Lemma 8: If there exists a singleton attractor for N , in its initial state, whenever the values of one $q'_i q_i$ are 10, there is exact one pair of vertices $q'_j q_j = 01$

Proof. First, $q_i(t+1) = 0$ when $q'_i q_i = 10$. To make $q'_i(t+1) = 1$, the clause $(\bigvee_{j \neq i} q_j)$ should be 1, which means there must exist at least one vertex, say $q_j = 1$. The value of q'_j cannot be 1 according to Lemma 5, so $q'_j q_j = 01$. There would not be another such pair of vertices according to Lemma 7. $\square$

Lemma 9: If there exists a singleton attractor for N , in its initial state, there exists a clique of size k in G .

Proof. According to Lemma 4, if there exists a singleton attractor for N , in its initial state, there should exist at least one clause $(q_i \wedge (\bigwedge_{p \in V_c, (p,q) \notin E_c} \bar{p}_i \wedge \bar{p}'_i)) = 1$ for every $1 \leq i \leq k$, and the values of the vertices q_i s in such clauses should be 1. The values of $q'_i q_i$ for such vertices should be 01 according to Lemma 5. In addition, according to Lemma 7, for each $q \in V_c$, there is at most one vertex with $q'_i q_i = 01$. Therefore, $X_1, X_2, \dots, X_k$ can represent k disjoint non-empty subsets of V_c . The assignment $q'_i q_i = 01$ means that the vertex q is in the k -th subset.

Pick one vertex for each subset and there will be k vertices. Any pair of such vertices p and q , which are in the j -th and the i -th subset respectively (i.e. $p'_j p_j = 01$ and $q'_i q_i = 01$), should be connected with each other; otherwise, according to Lemma 7, $p'_i p_i = 10$ and the value of clause $(\bigwedge_{p \in V_c, (p,q) \notin E_c} \bar{p}_i \wedge \bar{p}'_i)$ in f_{X_i} equals 0, which violates the assumption. Therefore, the induced subgraph of the k vertices

is a clique of size k . $\square$

Theorem 10: The detection of a singleton attractor of a BN with bounded treewidth k is W[1]-hard for parameter k .

Proof. Given a k -clique problem, we can construct a BN N that is a partial $3k - 1$ -tree. According to Lemma 3 and Lemma 9, we know that the k -clique problem can be reduced to the detection of an SA on N . Since the k -clique problem is a W[1]-hard for parameter k , so is the problem of detecting an SA on N . $\square$

For the k -clique problem, Chen et al. [2] gave a tight lower bound of complexity; i.e., k -clique cannot be solved in $n^{o(k)}$ unless all SNP problems (defined in [10]) are solvable in subexponential time, which seems unlikely. Therefore, under the same assumption, the problem of SA detection of a BN with bounded treewidth k cannot be solved in $n^{o(k)}$ time. The best known algorithm for detecting k -clique runs in time $O(n^{0.792k})$ [10], which means that, given a spacial BN in the reduction (a partial $3k - 1$ -tree equivalent to a k -clique problem), SA detection can be done in polynomial time if k is a constant. On the other hand, for more complex BNs, i.e. when a Boolean function is allowed to represent a SAT problem, SA detection of a BN is NP-hard even for a partial-1 tree. One of our future works for SA detection of a BN with bounded treewidth k is to prove the W[1]-hardness for k with more restricted class of BN.

Another possible future work might be the application of the notion of nice tree decomposition. It has been utilized to improve the fixed-parameter algorithms of problems such as DOMINATING SET in a partial k -tree [14].

6. Conclusion

In this article, we gave a fixed-parameter algorithm for detecting an SA in an AND/OR BN with bounded treewidth k . We also extended the algorithm for detecting an SA in a constant-depth NC-BN with bounded treewidth. However, there is not plausible fixed-parameter algorithm for detecting an SA in a general BN with bounded treewidth, since we proved the fixed-parameter intractability of this problem. It is still unknown that the problem of detecting an SA in an NC-BN with bounded treewidth is fixed parameter tractable or W[1]-hard.

Acknowledgements

CJC acknowledges the financial support from Interchange Association, Japan (IAJ) and National Science Council (NSC), Taiwan for providing a Summer Program grant. TT was partially supported by JSPS, Japan (Grant-in-Aid for Young Scientists (B) 25730005). KMC and CJC were also supported in part by NSC grants 100-2221-E-002-131-MY3 and 101-2221-E-002-063-MY3. TA was partially supported by Grant-in-Aid #22240009 from JSPS, Japan.

References

- [1] T. Akutsu, S. Kosub, A.A. Melkman, and T. Tamura, "Finding a periodic attractor of a Boolean network," *IEEE/ACM Trans. Computational Biology and Bioinformatics*, vol.9, pp.151–160, 2012.
- [2] J. Chen, B. Chor, M. Fellows, X. Huang, D. Juedes, I. Kanj, and G. Xia, "Tight lower bounds for certain parameterized NP-hard problems," *Proc. 19th IEEE Annual Conference on Computational Complexity*, pp.150–160, 2004.
- [3] V. Devloo, P. Hansen, and M. Labbé, "Identification of all steady states in large networks by logical analysis," *Bulletin of Mathematical Biology*, vol.65, pp.1025–1051, 2003.
- [4] E. Dubrova and M. Teslenko, "A SAT-based algorithm for finding attractors in synchronous Boolean networks," *IEEE/ACM Trans. Computational Biology and Bioinformatics*, vol.8, pp.1393–1398, 2011.
- [5] J. Flum and M. Grohe, *Parameterized Complexity Theory*, Springer, 2006.
- [6] A. Garg, A. DiCara, I. Xenarios, L. Mendoza, and G. DeMichel, "Synchronous versus asynchronous modelling of gene regulatory networks," *Bioinformatics*, vol.24, pp.1917–1925, 2008.
- [7] A.S. Jarrah, B. Raposa, and R. Laubenbacher, "Nested analyzing, unate cascade, and polynomial functions," *Physica D*, vol.233, pp.167–174, 2007.
- [8] S.A. Kauffman, *The Origins of Order: Self-organization and Selection in Evolution*, Oxford Univ. Press, New York, 1993.
- [9] A.A. Melkman, T. Tamura, and T. Akutsu, "Determining a singleton attractor of an AND/OR Boolean network in $O(1.587^n)$ time," *Inf. Process. Lett.*, vol.110, pp.565–569, 2010.
- [10] J. Nešetřil and S. Poljak, "On the complexity of the subgraph problem," *Commentationes Mathematicae Universitatis Carolinae*, vol.26, no.2, pp.415–419, 1985.
- [11] H. Tamaki, "A directed path-decomposition approach to exactly identifying attractors of Boolean networks," *Proc. 10th International Symposium on Communications and Information Technologies*, pp.844–849, 2010.
- [12] T. Tamura and T. Akutsu, "Detecting a singleton attractor in a Boolean network utilizing SAT algorithms," *IEICE Trans. Fundamentals*, vol.E92-A, pp.493–501, 2009.
- [13] H.L. Bodlaender, "A linear time algorithm for finding tree-decompositions of small treewidth," *Proc. 25th Annual ACM Symposium on Theory of Computing. ACM*, pp.226–234, 1993.
- [14] J. Alber and R. Niedermeier, "Improved tree decomposition based algorithms for domination-like problems," *LATIN 2002: Theoretical Informatics*, pp.613–627, Springer Berlin Heidelberg, 2002.

Chia-Jung Chang got his B.S. and M.S. degrees in computer science from National Taiwan University in 2004 and 2006, respectively. He worked as a software engineer in Foxconn Technology Group from 2007 to 2009. He is currently a Ph.D. student in Department of Computer Science and Information Engineering, National Taiwan University. His research interests include bioinformatics and algorithms.

Takeyuki Tamura received B.E., M.E. and Ph.D. degrees in informatics from Kyoto University, Japan, in 2001, 2003, and 2006, respectively. He joined Bioinformatics Center, Institute for Chemical Research, Kyoto University as a postdoctoral fellow in April, 2006. He works as an assistant professor from December, 2007. His research interests are bioinformatics and algorithms.

Kun-Mao Chao was born in Tou-Liu, Taiwan, in 1963. He received the B.S. and M.S. degrees in computer engineering from National Chiao-Tung University, Taiwan, in 1985 and 1987, respectively, and the Ph.D. degree in computer science from The Pennsylvania State University, University Park, in 1993. He is currently a professor of computer science and bioinformatics at National Taiwan University, Taipei, Taiwan. From 1987 to 1989, he served in the ROC Air Force Headquarters as a system engineer.

From 1993 to 1994, he worked as a postdoctoral fellow at Penn State's Center for Computational Biology. In 1994, he was a visiting research scientist at the National Center for Biotechnology Information, National Institutes of Health, Bethesda, Maryland. Before joining the faculty of National Taiwan University, he taught in the Department of Computer Science and Information Management, Providence University, Taichung, Taiwan, from 1994 to 1999, and the Department of Life Science, National Yang-Ming University, Taipei, Taiwan, from 1999 to 2002. He was a teaching award recipient of both Providence University and National Taiwan University. His current research interests include algorithms and bioinformatics. He is a member of Phi Tau Phi and Phi Kappa Phi.

Tatsuya Akutsu received his M.Eng. degree in Aeronautics in 1986 and a Dr. Eng. degree in Information Engineering in 1989 both from The University of Tokyo, Japan. From 1989 to 1994, he was with Mechanical Engineering Laboratory, Japan. He was an associate professor in Gunma University from 1994 to 1996 and in Human Genome Center, The University of Tokyo from 1996 to 2001 respectively. He joined Bioinformatics Center, Institute for Chemical Research, Kyoto University, Japan as a professor in Oct. 2001. His research interests include bioinformatics and discrete algorithms.

His research interests include bioinformatics and discrete algorithms.