数理解析研究所講究録
第1644巻 2009年 110-119

Detection and Enumeration of Steady States in Biological Information Networks

京都大学・化学研究所・バイオインフォマティクスセンター　阿久津　達也 (Tatsuya Akutsu)

Bioinformatics Center, Institute for Chemical Research, Kyoto University

Abstract

Detection and enumeration of steady-states in biological information networks is important in bioinformatics and systems biology. In this article, we focus on the Boolean network and review algorithms developed by the author and collaborators, where the Boolean network is known as a mathematical model of a genetic network.

1 Introduction

Analysis of various kinds of biological information networks is an important topic in bioinformatics, computational biology, and systems biology. In order to analyze these networks, various kinds of mathematical models have been proposed. Among them, the Boolean network (BN, in short) has received much attention [11]. BN is a model of genetic networks and is quite simple: each node corresponds to a gene and takes either 0 (inactive) or 1 (active), and the states of nodes change synchronously according to regulation rules given as Boolean functions. Thus, a BN with n nodes has a total of 2^n possible global states. Since each global state transits to one global state, beginning from any initial global state, the system will eventually evolve into a limited set of stable states called *attractors*. An attractor consisting of only one state is called a *singleton attractor* or a *fixed point*. Otherwise, it is called a *cyclic attractor*.

Since attractors are associated to distinct cell states, extensive studies have been done on the distribution of attractors [6, 11, 17]. In particular, many studies have been done on the average case numbers and lengths of attractors in BNs with maximum or average indegree K (i.e., each node has K incoming edges at the maximum or on the average), where the assumption of the indegree is reasonable because most nodes in real genetic networks do not have large indegrees. However, no conclusive results have not yet been obtained.

From a computational viewpoint, not so much attention had been paid for detection and/or enumeration of attractors. However, maybe due to the need for analyzing models of real genetic networks, extensive studies have recently been done on detecting and enumerating attractors. Akutsu et al. showed that deciding existence of a singleton attractor is NP-complete and counting the number of singleton attractors is #P-complete [1]. They also proposed an algorithm for enumerating all singleton attractors using a feedback vertex set [1]. Tošić showed that the counting problem remains #P-complete even if graphs are restricted to be planer bipartite graphs [23]. Kosub showed that the existence problem can be solved in polynomial time for several special cases relating with bounded treewitdth [12]. Several heuristic methods have also been proposed for enumeration of fixed points and/or cyclic attractors [5, 7, 9].

In this article, we briefly review our recent results on detection and enumeration of attractors. After introducing BN and the attractor problems, we review SAT-based algorithms proposed in [3], [19] and [21], where SAT denotes the Boolean satisfiability problem. As an extension of the simple reduction of the singleton attractor detection problem to SAT [19], we newly provide a reduction from the cyclic attractor detection problem to SAT, which gives an $o(2^n)$ time algorithm for the cyclic attractor detection problem for fixed period and K. We also review branch-and-bound type algorithms for enumerating singleton and small cyclic attractors [25]. Though these algorithms are simple, they work much faster than the naive algorithm both in theory (in the sense of the average case time complexity) and in practice. Finally, we conclude with future work.

2 Preliminaries

2.1 Boolean Network

A BN is represented by a set of *nodes* and a set of regulation rules for nodes, where each node corresponds to a gene when BN is regarded as a model of a genetic network. Each node takes either 0 or 1 at each discrete time t: 1 (resp. 0) means that the corresponding gene is active (resp. inactive) at time t. A regulation rule for each node is given in the form of a Boolean function and the states of nodes change synchronously. An example is given in Fig. 1. In this example, the state of node v_1 at time $t+1$ is determined by the state of node v_3 at time t. The states of node v_2 and v_3 at time $t+1$ are determined by logical AND of the state of node v_1 and negation of the state of node v_3 at time t and by logical AND of the state of node v_1 and negation of the state of node v_2 at time t, respectively. We use $x \wedge y$, $x \vee y$, $x \oplus y$, $\overline{x}$ to denote logical AND of x and y, logical OR of x and y, exclusive OR of x and y, and logical NOT of x, respectively. Dynamics of a BN is well-described by a *state transition table* and a *state transition diagram* shown in Fig. 1. For example, the fourth row of the table means that if the state of BN is $[0,1,1]$ at time t then the state will be $[1,0,0]$ at time $t+1$, and the arc from 111 to 100 in the diagram means that if the state of BN is $[1,1,1]$ at time t the state will be $[1,0,0]$ at time $t+1$.

Now we will give a formal definition of BN. A *Boolean network* $G(V,F)$ consists of a set $V = \{v_1, \ldots, v_n\}$ of nodes and a list $F = (f_1, \ldots, f_n)$ of *Boolean functions*, where a Boolean function $f_i(v_{i_1}, \ldots, v_{i_k})$ with inputs from specified nodes $v_{i_1}, \ldots, v_{i_k}$ is assigned to each node v_i. We use $IN(v_i)$ to denote the set of input nodes $v_{i_1}, \ldots, v_{i_k}$ to v_i. Each node takes either 0 or 1 at each discrete time t, and the state of node v_i at time t is denoted by $v_i(t)$. Then, the state of node v_i at time $t+1$ is determined by

$$v_i(t+1) = f_i(v_{i_1}(t), \ldots, v_{i_{k_i}}(t)).$$

Here we let $\mathbf{v}(t) = [v_1(t), \ldots, v_n(t)]$, which is called a *Gene Activity Profile* (GAP) or a *global state* at time t. We also write $v_i(t+1) = f_i(\mathbf{v}(t))$ and $\mathbf{v}(t+1) = \mathbf{f}(\mathbf{v}(t))$ to denote the regulation rule for v_i and the regulation rule for the whole BN, respectively. We define the set of edges E by $E = \{(v_{i_j}, v_i) | v_{i_j} \in IN(v_i)\}$. Then, $G(V,E)$ is a directed graph representing the network topology of a BN. The number of input nodes to v_i is called the *indegree* of v_i. We use K to denote the *maximum indegree* of a BN, which strongly affects the computational complexities of various algorithms.

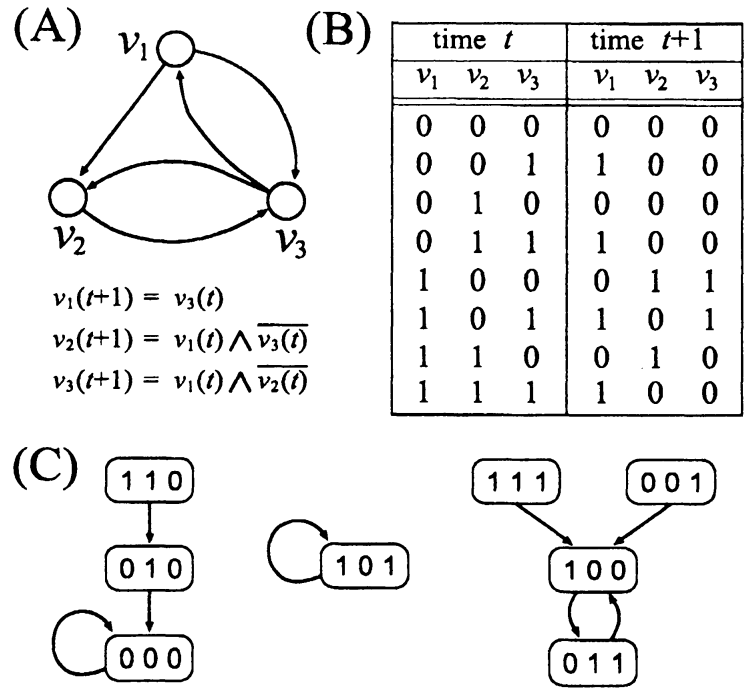

Figure 1: Example of a Boolean network. Dynamics of BN (A) is well-described by a state transition table (B) and by a state transition diagram (C).

2.2 Attractor

Starting from an initial GAP $\mathbf{v}(0)$, a BN will eventually reach a set of global states, called an *attractor*, which is a directed cycle in the state transition diagram. An attractor consisting of only one global state (i.e., $\mathbf{v} = \mathbf{f}(\mathbf{v})$) is called a *singleton attractor* or a *fixed point*. Otherwise, it is called a *cyclic attractor* with period p if it consists of p global states $\{\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^p\}$ (i.e., $\mathbf{v}^1 = \mathbf{f}(\mathbf{v}^p) = \mathbf{f}(\mathbf{f}(\mathbf{v}^{p-1})) = \cdots = \mathbf{f}(\mathbf{f}(\cdots \mathbf{f}(\mathbf{v}^1) \cdots))$, and $\mathbf{v}^1 \neq \mathbf{v}^q$ for all $q \neq 1$). The set of all GAPs that eventually evolve into the same attractor is called the *basin of attraction*. Different basins of attraction correspond to different connected components in the state transition diagram, and each connected component contains exactly one directed cycle. For example, in Fig. 1, 000 and 101 are singleton attractors, $\{011, 100\}$ is a cyclic attractor with period 2, and $\{001, 011, 100, 111\}$ is the basin of attraction for the attractor $\{011, 100\}$.

On detection and enumeration of attractors, we consider the following four problems:

Singleton attractor detection: Given a BN, decide whether or not there exists a singleton attractor,

Singleton attractor enumeration: Given a BN, output all singleton attractors,

Cyclic attractor detection: Given a BN and a period p, decide whether or not there exists a cyclic attractor with period p,

Cyclic attractor enumeration: Given a BN and a period p, output all cyclic attractors with period p.

3 SAT-based Algorithms

Since BN is a logic-based system, it is reasonable to try to apply existing algorithms developed for predicate logic to attractor detection and enumeration problems. In particular, it would be useful to apply algorithms for SAT because a number of algorithms have been developed for solving SAT [4, 10, 13, 16, 24]. In this section, we review SAT-based algorithms for detection and enumeration of singleton and small attractors in BN.

3.1 Simple Reduction to SAT

The singleton attractor detection problem for BNs with maximum indegree K can be transformed into $(K+1)$-SAT with n variables [19] in a simple manner. Recall that K-SAT is, given a set of clauses (i.e., a set of disjunctions of literals) over a set of Boolean variables, to decide whether or not there exists a 0-1 assignment to variables that satisfies all the clauses, where each clause consists of at most K literals.

Here, we only show a reduction procedure for the cases of BNs with maximum indegree 2, from which extensions to the other cases are straight-forward.

Let $\mathbf{v}$ be a GAP of a BN. Recall that $\mathbf{v}$ is a singleton attractor if $v_i = f_i(\mathbf{v})$ holds for all $i = 1, \ldots, n$, where we also use v_i to denote the state of v_i in a singleton attractor. In the following, l_i denotes either v_i or $\overline{v_i}$. We begin with the empty set. For $i = 1$ to n, we add clause(s) to the set according to the following rules:

$$
\begin{aligned}
v_i = l_j \vee l_k &\iff (\overline{v_i} \vee l_j \vee l_k) \wedge (v_i \vee \overline{l_j \vee l_k}) \\
&\iff (\overline{v_i} \vee l_j \vee l_k) \wedge (v_i \vee (\overline{l_j} \wedge \overline{l_k})) \\
&\iff (\overline{v_i} \vee l_j \vee l_k) \wedge (v_i \vee \overline{l_j}) \wedge (v_i \vee \overline{l_k}), \\
v_i = l_j \wedge l_k &\iff (\overline{v_i} \vee (l_j \wedge l_k)) \wedge (v_i \vee (\overline{l_j \wedge l_k})) \\
&\iff (\overline{v_i} \vee l_j) \wedge (\overline{v_i} \vee l_k) \wedge (v_i \vee \overline{l_j} \vee \overline{l_k}), \\
v_i = l_j \oplus l_k &\iff (\overline{v_i} \vee ((l_j \vee l_k) \wedge (\overline{l_j} \vee \overline{l_k})) \wedge (v_i \vee \overline{(l_j \vee l_k) \wedge (\overline{l_j} \vee \overline{l_k})}) \\
&\iff (\overline{v_i} \vee l_j \vee l_k) \wedge (\overline{v_i} \vee \overline{l_j} \vee \overline{l_k}) \wedge (v_i \vee \overline{(l_j \vee l_k)} \vee \overline{(\overline{l_j} \vee \overline{l_k})}) \\
&\iff (\overline{v_i} \vee l_j \vee l_k) \wedge (\overline{v_i} \vee \overline{l_j} \vee \overline{l_k}) \wedge (v_i \vee \overline{l_j} \vee l_k) \wedge (v_i \vee l_j \vee \overline{l_k}).
\end{aligned}
$$

Then, it is seen that a regulation rule for a node v_i is transformed into at most four clauses in 3-SAT. It is to be noted that any Boolean function with two inputs can be represented by $v_i = 0$, $v_i = 1$, $v_i = l_j$, or one of the above rules, where the cases of $v_i = 0$, $v_i = 1$ and $v_i = l_j$ can also be transformed into 3-SAT clauses. Thus, the singleton attractor detection problem for BNs with maximum indegree 2 is reduced to 3-SAT with n variables and at most $4n$ clauses. By extending it to general K, we have the following.

Proposition 1 *[19] Any instance of the singleton attractor detection problem for a BN of maximum indgree K with n nodes can be reduced in polynomial time to an instance of $(K+1)$-SAT with at most $2^{K+1} \cdot n$ clauses and n variables.*

This result can be extended for the cyclic attractor detection problem with period p. For simplicity, we consider the case of $p = 2$. Then, it is straight-forward to see that $\{\mathbf{v}, \mathbf{f}(\mathbf{v})\}$ is a cyclic attractor of period 2 if and only if $\mathbf{v} = \mathbf{f}(\mathbf{f}(\mathbf{v}))$ and $\mathbf{v} \neq \mathbf{f}(\mathbf{v})$ hold. Furthermore, we can see that the first condition for a given instance can be represented by an instance of

(K^2+1)-SAT with at most $2^{K^2+1} \cdot n$ clauses and n variables, and the second condition can be represented by an instance of $(K+1)$-SAT with at most $2^{K+1} \cdot n$ clauses and n variables. By extending it to general p, we have the following.

Theorem 1 *Any instance of the cyclic attractor detection problem with period p for a BN of maximum indgree K with n nodes can be reduced in polynomial time to an instance of (K^p+1)-SAT with at most $(\sum_{p=1}^{K} 2^{Kp+1}) \cdot n$ clauses and n variables.*

Combining with this result with $o(2^n)$ time algorithms for K-SAT [4], we can see that the cyclic attractor detection problem can be solved in $o(2^n)$ time for fixed p and K. Furthermore, we can see that the above mentioned reduction preserves the number of solutions. Therefore, counting the number of cyclic attractors might be done in $o(2^n)$ time for fixed p and K using algorithms for $\#K$-SAT problems [13].

3.2 Combination of SAT with Other Techniques

SAT algorithms can be used for developing algorithms for other special cases of BNs. Indeed, we considered AND/OR BNs, in which each Boolean function is limited to AND or OR of literals whereas there is no restriction on the maximum indgree [19, 21]. We developed algorithms for the singleton attractor detection problem for these AND/OR BNs. Here, we briefly review the basic idea used in these algorithms.

Suppose that the following function is assigned to a node v_i:

$$v_i(t+1) = v_1(t) \wedge v_2(t) \wedge \cdots \wedge v_h(t).$$

Consider four possible assignments for (v_1, v_i): $(0,0)$, $(0,1)$, $(1,0)$ and $(1,1)$. Among these assignments, three satisfy the condition of a singleton attractor whereas one (i.e., $(0,1)$) does not satisfy it. Therefore, by examining three assignments, we can eliminate two nodes. If we could continue this procedure until there is no remaining node, we would have the complexity of $O(3^{(n/2)}) \approx O(1.733^n)$ by solving

$$g(2) = 3, \quad g(k) = 3 \cdot g(k-2).$$

However, we cannot continue the above mentioned procedure until there is no remaining node. Therefore, we developed the following algorithm [19].

1. Let all the nodes be non-assigned.
2. While there exists a non-assigned node pair $(u, v) \in E$, examine all possible 3 assignments on $(u.v)$ recursively.
3. Let U be the set of nodes whose values were already assigned.
4. If $|U| > \alpha n$, examine all possible assignments on the remaining nodes and then check the condition of a singleton attractor. Otherwise, compute an appropriate assignment using [24] and then check the condition of a singleton attractor.

By letting $\alpha = 0.767$, we proved that this algorithm works in $O(1.792^n)$ time [19]. By combining with some other techniques, we obtained the following result.

Theorem 2 *[21] The singleton attractor detection problem for AND/OR BNs can be solved in $O(1.757^n)$.*

We also considered a special case of AND/OR BNs in which $G(V, E)$ is planar. In this case, by utilizing the *planar separator theorem* [14], we showed that the singleton attractor detection problem for planar AND/OR BNs can be solved in $O((1+\epsilon)^n)$ time where ϵ is an arbitrarily small positive constant.

In Section 3.1, we showed that the singleton attractor detection problem for BNs with maximum indegree K can be reduced to $(K+1)$-SAT. However, it is natural to ask whether or not there exist better algorithms. For the case of $K = 2$ (which remains NP-complete [19]), we showed that the answer is YES. By extending the idea mentioned in the first part of this subsection and combining it with the fastest 3-SAT algorithm [16], we developed an algorithm [3] that is very slightly faster than the simple reduction based algorithm (which also uses [16]).

4 Simple Recursive Algorithms and Their Average Case Analyses

Though SAT-based algorithms might be useful for the singleton attractor detection problems, these might not be so useful for the singleton attractor enumeration problems. Furthermore, we need to use existing SAT solvers in practice and it may be difficult to customize them in order to cope with some other constraints. Therefore, we developed several algorithms for enumerating singleton attractors and cyclic attractors with short periods [25], which do not use SAT algorithms. In this section, we briefly review a basic version of these algorithms, which is referred to as the *basic recursive algorithm*.

The number of singleton attractors in a BN depends on the regulatory rules of the network. If the rules are $v_i(t+1) = v_i(t)$ for all i, the number of singleton attractors is 2^n. Thus, it would take at least $O(2^n)$ time in the worst case if all the singleton attractors are to be enumerated. On the other hand, it is known that the average number of singleton attractors is 1 regardless of n and K [8, 15]. The basic recursive algorithm was designed based on these facts. It examines much smaller number of GAPs than 2^n in the average case.

In the basic recursive algorithm, a partial GAP (i.e., $[v_1, \ldots, v_m]$ for $m < n$) is extended one by one towards a complete GAP (i.e., singleton attractor), according to a given ordering of nodes (i.e., a random gene ordering). As soon as it is found that a partial GAP cannot be extended to a singleton attractor, the next partial GAP is examined. The pseudocode of this algorithm is given below, where it is invoked with $m = 1$.

Procedure *EnumerateSingletonAttractor*(v, m)
  if $m = n + 1$
  then Output $[v_1, v_2, \cdots, v_n]$ and **return**
  for $b = 0$ **to** 1 **do**
    $v_m := b$
    if it is found that $f_i(\mathbf{v}) \neq v_i$ for some $i \leq m$
    then continue
    else *EnumerateSingletonAttractor*$(v, m+1)$
  return

This algorithm extends a partial GAP by one node at a time in a recursive manner. At the m-th recursive step, the states of the first $m-1$ nodes (i.e., a partial GAP) are already determined. Then, the algorithm extends the partial GAP by letting $v_m = 0$. If $f_i(\mathbf{v}) = v_i$ holds or the value of v_i is not determined for each $i = 1, \ldots, m$ (i.e., there is a possibility that

Table 1: Average case time complexities of basic, outdegree-based, and BFS-based algorithms for the singleton attractor detection problem [25].

K	2	3	4	5	6
basic	1.35^n	1.43^n	1.49^n	1.53^n	1.57^n
outdegree-based	1.19^n	1.27^n	1.34^n	1.41^n	1.45^n
BFS-based	1.16^n	1.27^n	1.35^n	1.41^n	1.45^n

the current partial GAP can be extended to a singleton attractor), the algorithm proceeds to the next recursive step. Otherwise, it modifies the partial GAP by letting $v_m = 1$ and executes a similar procedure. After examining both $v_m = 0$ and $v_m = 1$, the algorithm returns to the previous recursive step. Since the number of singleton attractors is small in most cases, it is expected in most cases that the algorithm does not examine many partial GAPs with large m. The average case time complexity is estimated as follows [25].

Assume that we have tested the first m out of n nodes, where $m \geq K$. For all $i \leq m$, $f_i(\mathbf{v}) \neq v_i$ holds with probability

$$P(f_i(\mathbf{v}) \neq v_i) = 0.5 \cdot \frac{\binom{m}{k_i}}{\binom{n}{k_i}} \approx 0.5 \cdot (\frac{m}{n})^{k_i} \geq 0.5 \cdot (\frac{m}{n})^K,$$

where we assume that Boolean functions of k_i inputs are selected at uniformly random. If $f_i(\mathbf{v}) \neq v_i$ holds for some $i \leq m$, the algorithm cannot proceed to the next recursive level. Therefore, the probability that the algorithm examines the $(m+1)$-th node is no more than

$$[1 - P(f_i(\mathbf{v}) \neq v_i)]^m = [1 - 0.5 \cdot (\frac{m}{n})^K]^m.$$

Thus, the number of recursive calls executed for the first m nodes is at most

$$f(m) = 2^m \cdot [1 - 0.5 \cdot (\frac{m}{n})^K]^m.$$

Let $s = \frac{m}{n}$, and $F(s) = [2^s \cdot (1 - 0.5 \cdot s^K)^s]^n = [(2 - s^K)^s]^n$. Then, the average case time complexity of the algorithm can be estimated as $O((\max(g))^n)$ for fixed K, where $g(s) = (2 - s^K)^s$. By means of numerical calculation for $\max(g)$, we obtained the average case time complexities for $K = 2, \ldots, 6$ as in the first row of Table 1. It should be noted that the naive exhaustive search-based algorithm takes at least $O(2^n)$ time. Therefore, the basic recursive algorithm is much faster than the naive algorithm for small K.

We obtained variants of this basic recursive algorithm by sorting nodes before invoking the recursive procedure [25]. In particular, we used the orderings of nodes according to the outdegree and BFS (breadth-first search). For these algorithms, we obtained theoretical estimates of the average case time complexity as in Table 1. We also performed computational experiments to confirm these theoretical results (it is to be noted that some approximations were included in theoretical analyses). As a result, good agreement was observed. We also extended the basic recursive algorithm for enumerating cyclic attractors with short periods [25].

5 Concluding Remarks

In this article, we reviewed algorithms developed by the author and collaborators for attractor detection and enumeration problems for BNs. We showed that the singleton and cyclic attractor detection problems for BNs with maximum indegree K are reduced to SAT problems. We also reviewed other types of algorithms for attractor detection problems for special cases of BNs. For the singleton and cyclic attractor enumeration problems, we reviewed simple recursive algorithms. None of the complexity results reviewed in this article was proven to be optimal. Therefore, improvements of the time complexities are left as open problems. Though we only considered attractor problems for BNs, there exist other computational problems on BNs [2]. In particular, control of BNs and more general models might be important because it might have applications to developments of novel drugs and treatment methods for difficult diseases.

In this article, BNs are considered as a model of genetic networks. However, BN and its variants can also be used as models of other types of biological information networks. Recently, BN-like models were proposed for modeling metabolic networks [18, 20], where chemical compounds are regarded as OR nodes and chemical reactions and/or enzymes are regarded as AND nodes. Using these models, problems of deciding the minimum number of chemical reactions/enzymes to be inactivated for preventing production of specified chemical compounds were studied, which may have applications to identification of multiple drug targets. We studied one formulation of these problems both from a theoretical viewpoint and from a practical viewpoint. From a theoretical viewpoint, we showed that the problem is NP-hard [22]. We also developed an $O(1.822^n)$ time algorithm for the case where the maximum indegree of reaction nodes is bounded by 2 [22]. From a practical viewpoint, we developed an algorithm using integer linear programming and feedback vertex sets [20].

We understand that there is a criticism that BN is too simple as a model of genetic networks, metabolic networks, and/or other types of biological information networks. However, studies on BNs may provide some insights into other models. At least, hardness results should hold for more general models. Some ideas in positive results might also be useful for design and analysis of algorithms for more general models. Therefore, extensions of BNs and development of algorithms for such extended models are important future work.

Acknowledgment

I would like to thank all collaborators on the works mentioned in this article.

References

[1] T. Akutsu, S. Kuhara, O. Maruyama, and S. Miyano, A system for identifying genetic networks from gene expression patterns produced by gene disruptions and overexpressions, *Genome Informatics*, vol.9, pp.151–160, 1998.

[2] T. Akutsu, M. Hayashida, and T. Tamura, Algorithms for inference, analysis and control of Boolean networks, *Proc. 3rd Int. Conf. Algebraic Biology*, pp.1–15, 2008.

[3] T. Akutsu and T. Tamura, On finding a fixed point in a Boolean network with maximum indegree 2, submitted.

[4] E. Dantsin, A. Goerdt, E. A. Hirsch, R. Kannan, J. M. Kleinberg, C. H. Papadimitriou, P. Raghavan, and U. Schöning, A deterministic $(2 - 2/(k + 1))n$ algorithm for k-SAT based on local search, *Theoretical Computer Science*, vol.289, pp.69–83, 2002.

[5] 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.

[6] B. Drossel, T. Mihaljev, and F. Greil, Number and length of attractors in a critical Kauffman model with connectivity one, *Physical Review Letters*, vol.94, 088701, 2005.

[7] A. Garg, A. DiCara, I. Xenarios, L. Mendoza, and G. DeMicheli, Synchronous versus asynchronous modeling of gene regulatory networks, *Bioinformatics*, vol.24, pp.1917–1925, 2008.

[8] I. Harvey and T. Bossomaier, Time out of joint: Attractors in asynchronous random boolean networks, *Proc. 4th European Conference on Artificial Life*, pp.67–75, 1997.

[9] D. J. Irons, Improving the efficiency of attractor cycle identification in Boolean networks, *Physica D*, vol.217, pp.7–21, 2006.

[10] K. Iwama and S. Tamaki, Improved upper bounds for 3-SAT, *Proc. 15th ACM-SIAM Symp. Discrete Algorithms*, pp.328–329, 2004.

[11] S. A. Kauffman, *The Origins of Order: Self-organization and Selection in Evolution*, Oxford Univ. Press, New York, 1993.

[12] S. Kosub, Dichotomy results for fixed-points existence problems for Boolean dynamical systems, *Mathematics in Computer Science*, vol.1, pp.487–505, 2008.

[13] K. Kutzkov, New upper bound for the #3-SAT problem, *Information Processing Letters*, vol.105, pp.1–5, 2007.

[14] R. J. Lipton and R. E. Tarjan, A separator theorem for planar graphs, *SIAM Journal on Applied Mathematics*, vol.36, pp.177–189, 1979.

[15] A. Mochizuki, An analytical study of the number of steady states in gene regulatory networks, *Journal of Theoretical Biology*, vol.236, pp.291–310, 2005.

[16] D. Rolf, Improved bound for the PPSZ/Schöning-algorithm for 3-SAT, *Journal of Satisfiability, Boolean Modeling and Computation*, vol.1, pp.111–122, 2006.

[17] B. Samuelsson and C. Troein, Superpolynomial growth in the number of attractors in kauffman networks, *Physical Review Letters*, vol.90, 098701, 2003.

[18] P. Sridhar, B. Song, T. Kahveci, and S. Ranka, Mining metabolic networks for optimal drug targets, *Proc. 13th Pacific Symposium on Biocomputing*, pp.291–302, 2008.

[19] T. Tamura and T. Akutsu, Detecting a singleton attractor in a Boolean network utilizing SAT algorithms, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol.E92-A, pp.493–501, 2009.

[20] T. Tamura, K. Takemoto, and T. Akutsu, Measuring structural robustness of metabolic networks under a Boolean model using integer programming and feedback vertex sets, *Proc. 2nd Int. Workshop on Intelligent Informatics in Biology and Medicine/ 3rd Int. Conf. Complex, Intelligent and Software Intensive Systems*, pp.819–824, 2009.

[21] T. Tamura and T. Akutsu, Algorithms for singleton attractor detection in planar and nonplanar AND/OR Boolean networks, *Mathematics in Computer Science*, to appear.

[22] T. Tamura, K. Takemoto, and T. Akutsu, On the complexity of measuring structural robustness of metabolic networks, manuscript.

[23] P. Tošić, On the complexity of counting fixed points and gardens of Eden in sequential dynamical systems on planar bipartite graphs, *International Journal of Foundations of Computer Science*, vol.17, pp.1179–1203, 2006.

[24] M. Yamamoto, An improved $\tilde{O}(1.234^m)$-time deterministic algorithm for SAT, *Proc. 16th International Symposium on Algorithms and Computation*, pp.644–653, 2005.

[25] S-Q. Zhang, M. Hayashida, T. Akutsu, W-K. Ching, and M. K. Ng, Algorithms for finding small attractors in Boolean networks, *EURASIP Journal on Bioinformatics and Systems Biology*, vol.2007, 20180, 2007.