IDENTIFICATION OF GENETIC NETWORKS FROM A SMALL NUMBER OF GENE EXPRESSION PATTERNS UNDER THE BOOLEAN NETWORK MODEL

Tatsuya AKUTSU, Satoru MIYANO
Human Genome Center, Institute of Medical Science,
University of Tokyo, 4-6-1 Shirokanedai, Minato-ku, Tokyo 108-8639, Japan

Satoru KUHARA
Graduate School of Genetic Resources Technology,
Kyushu University. 6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan

Liang, Fuhrman and Somogyi (PSB98, 18-29, 1998) have described an algorithm for inferring genetic network architectures from state transition tables which correspond to time series of gene expression patterns, using the Boolean network model. Their results of computational experiments suggested that a small number of state transition (INPUT/OUTPUT) pairs are sufficient in order to infer the original Boolean network correctly. This paper gives a mathematical proof for their observation. Precisely, this paper devises a much simpler algorithm for the same problem and proves that, if the indegree of each node (i.e., the number of input nodes to each node) is bounded by a constant, only $O(\log n)$ state transition pairs (from 2^n pairs) are necessary and sufficient to identify the original Boolean network of n nodes correctly with high probability. We made computational experiments in order to expose the constant factor involved in $O(\log n)$ notation. The computational results show that the Boolean network of size 100,000 can be identified by our algorithm from about 100 INPUT/OUTPUT pairs if the maximum indegree is bounded by 2. It is also a merit of our algorithm that the algorithm is conceptually so simple that it is extensible for more realistic network models.

1 Introduction

Inference of gene regulation mechanism from time series of gene expression patterns is getting more important especially due to the invent of DNA microarray technology [3,9,11]. Expression profiles of several thousands of genes are now being produced for further analyses.

Some methods have been proposed for the inference of gene regulation mechanism from time series of gene expression patterns. Arkin, Shen and Ross [2] proposed a statistical method using correlation matrices to infer chemical reaction networks from time series of measured concentration of species. Although they treated chemical reaction networks, they also suggested that their method might be applied to genetic networks. However, it seems difficult to apply their method to the inference of large scale networks. DeRisi, Iyer and Brown [3] inferred a metabolic pathway from gene expression patterns of

Saccharomyces cerevisiae obtained by using DNA microarrays. Yuh, Bolouri and Davidson [11] constructed a network model similar to the Boolean network model from time series of expression patterns relating to a sea urchin gene. However, their inference methods are not systematic or automatic.

On the other hand, some studies have been done on the inference of genetic networks from state transition data using the *Boolean network* model [7,10] and its variants [1,8]. In particular, Liang, Fuhrman and Somogyi [5] proposed an algorithm named REVEAL for inference of Boolean networks (corresponding to genetic networks) from state transition tables (corresponding to time series of gene expression patterns). REVEAL used information theoretic principles in oder to reduce the search space. They made some computational experiments on REVEAL. The results suggested that only a small number of state transition pairs (100 pairs from 10^{15}) were sufficient for inferring Boolean networks with 50 nodes (genes) whose *indegree* (the number of input nodes to a node) was bounded by 3.

Independently, we have investigated strategies for identifying genetic networks from gene expression patterns derived by gene disruptions and gene overexpressions using a Boolean network-like model [1]. In Akutsu *et al.* [1], we proved mathematically a lower bound and an upper bound of the number of expression patterns required to identify the network correctly. We have not assumed that time series of expression patterns are observable in the paper [1] and thus the derived bounds on experimental complexity are too high to be practical if it would be applied directly. However, the recent progress [3,9,11] of biotechnology is making it possible to observe time series of gene expression patterns. Therefore, in this paper, we mathematically study the number of gene expression patterns required to identify the genetic network using the Boolean network model.

The contribution of this paper is a simple algorithm for identifying the original Boolean network from the state transition pairs (i.e., INPUT/OUTPUT expression pattern pairs) and its mathematical analysis. Its usefulness in practice is also verified by computational experiments.

Our algorithm is much simpler than REVEAL [5] although the efficiency of time and memory space of our algorithm may be worse than REVEAL. The simplicity of this algorithm makes its mathematical analysis possible. Moreover, this algorithm can be modified for counting or enumerating the networks consistent with given examples (i.e., state transition pairs). We prove *mathematically* that $O(\log n)$ (precisely, $\Theta(\log n)$) transition pairs are necessary and sufficient for our algorithm to identify the original Boolean network of n nodes with a high probability if the maximum indegree is bounded by a constant and transition pairs are given uniformly randomly from 2^n possible pairs, where

$\log x$ means $\log_2 x$ throughout this paper. Note that, although a lot of studies have been done on the number of examples required to identify Boolean functions in Computational Learning Theory [4], such results do not seem to be directly applicable to the identification of the Boolean network.

In order to expose the constant factor involved in $O(\log n)$ notation, we made computational experiments on our algorithm. The computational experiments reveal that the constant hidden in $O(\log n)$ notation is practically small. For example, Boolean networks of bounded indegree 2 with $100,000$ nodes can be identified from only 100 random state transition pairs. In order to investigate more practical situations, we made computational experiments where state transition pairs are generated from *attractors* [7]. Although the number of pairs required for identification is increased, it is still proportional to $\log n$.

Although the real genetic networks are different from the Boolean networks, the theoretical and practical results in this paper may be extended for more realistic models. Since the proposed algorithm is conceptually very simple, it is highly extensible for various situations. Possible extensions are discussed in the final section.

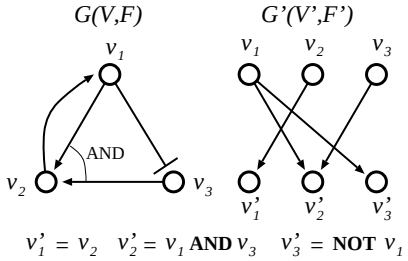
2 Boolean Network and Its Identification

2.1 Boolean Network

A *Boolean network* $G(V, F)$ consists of a set $V = \{v_1, \ldots, v_n\}$ of nodes representing genes 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. For a subset $U \subseteq V$, an *expression pattern* ψ of U is a function from U to $\{0, 1\}$. An expression pattern of V is also called a *state* of a Boolean network. That is, ψ represents the states of nodes (genes), where each node is assumed to take either 0 (not-express) or 1 (express) as its state value. If it does not cause confusion, we omit ψ. For example, we write $v_i = 1$ for denoting $\psi(v_i) = 1$. In a Boolean network, the expression pattern ψ_{t+1} at time $t + 1$ is determined by Boolean functions F from the expression pattern ψ_t at time t (i.e., $\psi_{t+1}(v_i) = f_i(\psi_t(v_{i_1}), \ldots, \psi_t(v_{i_k}))$).

It is convenient to consider a *wiring diagram* [5,7] $G'(V', F')$ of a Boolean network $G(V, F)$ (see Fig. 1). For each node v_i in V, let $v_{i_1}, \ldots, v_{i_k}$ be input nodes to v_i in $G(V, F)$. Then we consider an additional node v'_i, and we construct an edge from v_{i_j} to v'_i for each $1 \leq j \leq k$. Let $G'(V', F')$ the network with nodes $v_1, \ldots, v_n, v'_1, \ldots, v'_n$ constructed in this way. Then, the expression pattern of the set $\{v'_1, \ldots, v'_n\}$ is determined by $v'_i = f_i(v_{i_1}, \ldots, v_{i_k})$.

That is, the expression pattern of $\{v_1, \ldots, v_n\}$ corresponds to one at time t and the expression pattern of $\{v'_1, \ldots, v'_n\}$ corresponds to one at time $t+1$. Moreover, it is convenient to consider the expression pattern of $\{v_1, \ldots, v_n\}$ as the INPUT, and the expression pattern of $\{v'_1, \ldots, v'_n\}$ as the OUTPUT.



INPUT			OUTPUT		
v_1	v_2	v_3	v'_1	v'_2	v'_3
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

Figure 1: A Boolean network $G(V, F)$ and its wiring diagram $G'(V', F')$. In the (state transition) table, the INPUT column corresponds to the expression pattern (state) at time t and the OUTPUT column corresponds to the expression pattern (state) at time $t+1$.

2.2 Identification Problem

Next we formally define the *identification problem*. Relating to the identification problem, we also define the *consistency problem*, the *counting problem* and the *enumeration problem*.

Let (I_j, O_j) be a pair of expression patterns of $\{v_1, \ldots, v_n\}$, where I_j corresponds to the INPUT and O_j corresponds to the OUTPUT. We call the pair (I_j, O_j) an *example*.

We say that a node v_i in a Boolean network $G(V, F)$ is *consistent* with an example (I_j, O_j) if $O_j(v_i) = f_i(I_j(v_{i_1}), \ldots, I_j(v_{i_k}))$ holds. We say that a Boolean network $G(V, F)$ is *consistent* with (I_j, O_j) if all nodes are consistent with (I_j, O_j). For a set of examples $EX = \{(I_1, O_1), (I_2, O_2), \ldots, (I_m, O_m)\}$, we say that $G(V, F)$ (resp. node v_i) is *consistent* with EX if $G(V, F)$ (resp. node v_i) is consistent with all (I_j, O_j) for $1 \leq j \leq m$. Then, the problems are defined as follows (see also Fig. 2): **CONSISTENCY:** Given n (the number of nodes) and EX, decide whether or not there exists a Boolean network consistent with EX and output one if it exists; **COUNTING:** Given n and EX, count the number of Boolean networks consistent with EX; **ENUMERATION:** Given n and EX, output all the Boolean networks consistent with EX; **IDENTIFICATION:** Given n and EX, decide whether or not there exists a *unique* Boolean network consistent with EX and output it if it exists.

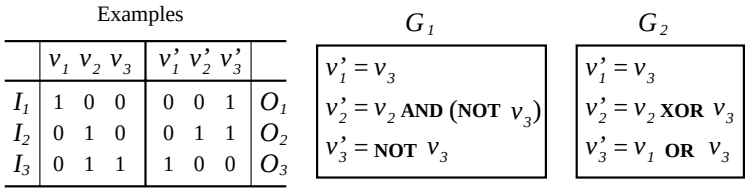


Figure 2: INPUT/OUTPUT expression patterns and Boolean networks. Boolean network G_1 is consistent with examples, while Boolean network G_2 is not consistent with the examples since node v_3 (in G_2) is not consistent with (I_3, O_3). In this case, the consistent Boolean network is not determined uniquely since we can obtain another consistent network by replacing $v_2' = v_2$ AND (NOT v_3) in G_1 with $v_2' = v_2$ XOR v_3.

3 Identification Algorithm

In this section, we only consider the Boolean network in which the *indegree* (i.e., the number of input nodes) of each node is bounded by a constant K, because it has been proved that exponentially many examples are required in order to identify input nodes to a high indegree node[1]. The importance of the constraint on the indegree is also pointed out in several papers[5,7].

Although we assume that the maximum indegree is bounded by K, the proposed algorithms can be applied to Boolean networks whose maximum indegree is not bounded. *In such a case, the algorithms correctly identify (or find) Boolean functions assigned to all nodes whose indegrees are at most K.*

3.1 Algorithms

In this subsection, for simplicity, we only show algorithms for the case of $K = 2$. But, they can be generalized to any K in a straightforward way.

First we show an algorithm for the *consistency problem* (see also Fig. 2). The algorithm below is natural and conceptually very simple since it simply outputs Boolean functions consistent with given examples.

(1) For each node $v_i \in V$, execute STEP (2).

(2) If there exists a triplet (f_i, v_k, v_h) satisfying $O_j(v_i) = f_i(I_j(v_k), I_j(v_h))$ for all $j = 1, \ldots, m$, output f_i as a Boolean function assigned to v_i and output v_k, v_h as input nodes to v_i.

In order to find a triplet (f_i, v_k, v_h), we use a simple exhaustive search: for each pair of nodes (v_k, v_h) $(k < h)$ and for each Boolean function f, we check whether or not $O_j(v_i) = f(I_j(v_k), I_j(v_h))$ holds for all j. For example, we consider the case of Fig. 2. In this case, for each v_i', we check all combinations

of $2^{2^2} = 16$ Boolean functions $f(x, y)$ (x AND y, x OR y, x AND (NOT y), $\cdots$) and 3 pairs of nodes $((v_1, v_2), (v_1, v_3), (v_2, v_3))$.

The above algorithm can be modified for other problems. For the *enumeration problem*, we replace STEP (2) with the following:

(2') Enumerate all triplets (f_i, v_k, v_h) satisfying $O_j(v_i) = f_i(I_j(v_k), I_j(v_h))$ for all $j = 1, \ldots, m$.

Then, any combination of triplets $((f_1, v_{k_1}, v_{h_1}), (f_2, v_{k_2}, v_{h_2}), \ldots, (f_n, v_{k_n}, v_{h_n}))$ can represent a consistent Boolean network. Of course, we carefully enumerate triplets since there exists more than two triplets which represent the same Boolean function (such as $v_k \wedge v_h$ and $v_h \wedge v_k$).

For the *counting problem*, we simply multiply the number of triplets consistent with each node.

For the *identification problem*, we replace STEP (2) with the following:

(2") If there exists only one triplet (f_i, v_k, v_h) satisfying $O_j(v_i) = f_i(I_j(v_k), I_j(v_h))$ for all $j = 1, \ldots, m$, output f_i as a Boolean function assigned to v_i and output v_k, v_h as input nodes to v_i.

3.2 Analysis

First we consider the time complexity of the algorithm for the consistency problem. There are 2^{2^K} Boolean functions with K input variables [7]. There are $\binom{n}{K}$ (possible) combinations of input nodes per node. Therefore, for each node, $O(2^{2^K} \cdot n^K)$ triplets are examined in the algorithm. For each triplets, m examples are examined. Therefore, $O(2^{2^K} \cdot n^K \cdot n \cdot m)$ pairs of Boolean functions and examples are examined in total. In order to examine one pair, $O(K)$ time is required. Therefore, the algorithm works in $O(K \cdot 2^{2^K} \cdot n^{K+1} \cdot m)$ time. Thus, the algorithm works in polynomial time for fixed K.

Similarly, we can show that the algorithms for the counting problem and the identification problem work in polynomial time for fixed K.

Theorem 1. The consistency problem, the counting problem and the identification problem can be solved in polynomial time for Boolean networks whose maximum indegrees are bounded by a constant.

Note that the enumeration problem can not be solved in polynomial time because the number of consistent Boolean networks may become exponential.

Next we analyze the number of INPUT/OUTPUT pairs required to identify the Boolean network uniquely. The following proposition was obtained directly from Theorem 3.2 in our previous paper [1] (see also Fig. 3).

Proposition 1. If all assignments (i.e., 2^{2K} assignments) of Boolean values to all subsets of V with $2K$ nodes (i.e., $\binom{n}{2K}$ subsets) appear in INPUT expression patterns, the Boolean function together with input nodes for each node is determined uniquely, if it exists.

Next we prove the main theorem.

Theorem 2. If $O(2^{2K} \cdot (2K+\alpha) \cdot \log n)$ INPUT expression patterns are given uniformly randomly, the following holds with probability at least $1 - \frac{1}{n^\alpha}$: there exists at most one Boolean network of n nodes with maximum indegree $\leq K$ which is consistent with given INPUT/OUTPUT pairs.

(Proof) We derive the number of INPUT expression patterns satisfying the condition of Proposition 1. For that purpose, we consider the probability that the condition is not satisfied when m random INPUT expression patterns are given.

For any fixed set of nodes $\{v_{i_1}, \ldots, v_{i_{2K}}\}$, the probability that a sub-assignment $v_{i_1} = v_{i_2} = \cdots = v_{i_{2K}} = 1$ does not appear in one random INPUT expression pattern is $1 - \frac{1}{2^{2K}}$. Thus, the probability that $v_{i_1} = \cdots = v_{i_{2K}} = 1$ does not appear in any of m random INPUT expression patterns is $(1 - \frac{1}{2^{2K}})^m$. Since the number of combinations of $2K$ nodes is less than n^{2K}, the probability that there exists a combination of $2K$ nodes for which an assignment $v_{i_1} = \cdots = v_{i_{2K}} = 1$ does not appear in any of m random INPUT expression patterns is at most $n^{2K} \cdot (1 - \frac{1}{2^{2K}})^m$. Since there are 2^{2K} possible assignments to $2K$ variables, the probability that the condition of Proposition 1 is not satisfied is at most $2^{2K} \cdot n^{2K} \cdot (1 - \frac{1}{2^{2K}})^m$. It is not difficult to see that $2^{2K} \cdot n^{2K} \cdot (1 - \frac{1}{2^{2K}})^m < p$ holds for $m > \ln 2 \cdot 2^{2K} \cdot (2K + 2K \log n + \log \frac{1}{p})$. Letting $p = \frac{1}{n^\alpha}$, we obtain the theorem. □

	EX_1				EX_2			
	v_1	v_2	v_3	v_4	v_1	v_2	v_3	v_4
I_1	0	1	1	1	1	1	0	0
I_2	1	0	1	1	0	0	1	1
I_3	1	1	0	1	1	0	1	0
I_4	1	1	1	0	0	1	0	1
I_5	0	0	0	0	1	1	1	1

Figure 3: Let $n = 4$ and $K = 1$. The condition of Proposition 1 is satisfied by EX_1, but not satisfied by EX_2 because a sub-assignment $\langle v_2 = 0, v_3 = 0 \rangle$ does not appear in EX_2.

Note that the probability of Theorem 2 is computed amongst all possible INPUT expression patterns, not amongst all Boolean networks. Therefore, for any Boolean network of fixed K, $O(\log n)$ INPUT/OUTPUT pairs are sufficient with high probability.

Theorem 2 seems surprising because only $O(\log n)$ INPUT/OUTPUT pairs are sufficient for the identification (for fixed K) although there are 2^n different INPUT expression patterns for a Boolean network. We can also prove that, *in the average*, $O(\log n)$ expression patterns are sufficient, where we omit the proof here.

3.3 Information Theoretic Lower Bound

Next we show an information theoretic lower bound on the number of INPUT/OUTPUT pairs required to identify the Boolean network uniquely.

Theorem 3. $\Omega(2^K + K \log n)$ INPUT/OUTPUT pairs are necessary in the worst case to identify the Boolean network of maximum indegree $\leq K$.
(Proof) We consider the number of mutually distinct Boolean networks. Since there are $\Omega(n^K)$ possible combinations of input nodes and 2^{2^K} possible Boolean functions per node, there are $\Omega((2^{2^K} \cdot n^K)^n)$ Boolean networks whose maximum indegree is at most K. Therefore, $\Omega(2^K n + nK \log n)$ bits are required to represent a Boolean network. On the other hand, information quantity obtained from one INPUT/OUTPUT pair is n bits. Therefore, $\Omega(2^K + K \log n)$ INPUT/OUTPUT pairs are required in the worst case. □

4 Computational Experiments

In Section 3, we proved a lower bound and an upper bound of the number of INPUT/OUTPUT pairs required to identify the Boolean networks. However, regarding to a constant factor depending on K, there is still a gap between them. Thus, in order to clarify the constant factor in a practical case, we have made computational experiments. We made a computer program using C language on SUN ULTRA ENTERPRISE-10000 with 64 processors and 16GB memory.

4.1 Random Expression Patterns

We first examine cases of $K = 2$ and $K = 3$ using randomly generated Boolean networks and randomly generated INPUT expression patterns.

For each n and each K, we randomly generate 10 Boolean networks, where each network is generated by, for each node, randomly choosing two input

nodes and a Boolean function from 2^{2^K} possible ones. INPUT expression patterns are generated randomly by independently assigning 1 to each node with probability 0.5. For each INPUT, OUTPUT is computed according to the Boolean functions assigned to the nodes. For each generated network, we generate INPUT/OUTPUT pairs until the network is identified uniquely and we count the number of INPUT/OUTPUT pairs. Since we use randomly generated INPUT expression patterns, we take the average number of 10 trials.

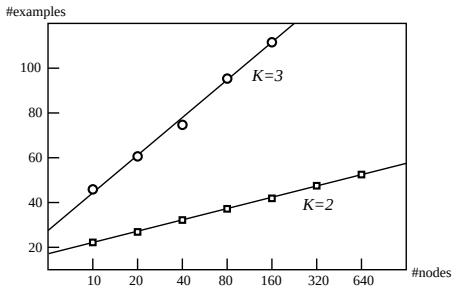


Figure 4: Results of computational experiments on the number of INPUT/OUTPUT pairs required to identify the Boolean networks of maximum indegree $K = 2, 3$. Note that X-axis is log-scaled and thus this graph shows that the number is proportional to $\log n$.

The results are shown in Fig. 4. In each case, it is seen that the number of INPUT/OUTPUT pairs is proportional to $\log n$. For the result of $K = 2$, it is seen that only less than 50 pairs are required to identify the network with 320 nodes. Since the number is proportional to $\log n$, this result suggests only 100 pairs are required to identify the network with $100,000$ ($\approx 320^2$) nodes in the case of $K = 2$. Even for $K = 3$, we can see that the number is not so large (< 300 pairs). From these results, it is seen that the constant factor on $\log n$ is not close to either the upper bound ($\ln 2 \cdot 2K \cdot 2^{2K}$) or the lower bound ($K$). It seems that the constant factor is near to $K \cdot 2^K$.

As for the CPU time for identification, it took less than 1 sec. for small n (e.g., $n \leq 40$), whereas it took more than 1 min. for large n (e.g., $n \geq 160$ and $K = 3$). So, we could not achieve enough computational experiments to derive the average number for the case of $n = 320$ and $K = 3$.

4.2 Expression Patterns Generated from Attractors

In the above, we use randomly generated INPUT/OUTPUT pairs. However, in real biological experiments, we observe expression patterns in a consecutive time sequence. In such a case, an output expression pattern at time t corre-

sponds to an input expression pattern for the next time step (i.e., $I_{t+1} = O_t$), and thus INPUT expression patterns can not be considered as random expression patterns.

Thus, in this experiment, we used only one randomly generated INPUT expression pattern I_1 and we used INPUT expression patterns generated by $I_{j+1} = O_j$ for $I_2, I_3, \ldots$ However, in this case, we could not identify the Boolean network uniquely because the INPUT expression pattern sequence fell into an *attractor*[7,10] and only a small number of different expression patterns appeared. We can not identify a Boolean function whose value does not change in an attractor cycle.

The above phenomenon is reasonable from a biological point of view. Attractors are considered as the target areas of an organism [5,7], e.g. cell types at the end of development, repaired tissue following a response to injury, or adaptation of metabolic gene expression following a change in nutrient environment in bacteria. Usually, it is not considered that all genes are always active. Indeed, there are some genes which are active only under some special environment (such as cell division or heat-shock). We can not discover the regulation mechanism for a gene whose expression pattern does not change in one environment. In such a case, we should observe gene expression patterns under other environments (or in other types of cells).

Thus, we generate another attractors if the network is not identified uniquely from expression patterns in one attractor. We use the following procedure to generate expression patterns: **(1)** Generate random expression pattern I_1, and generate INPUT expression patterns by $I_{j+1} = O_j$ until the same INPUT expression pattern as the previous INPUT expression pattern appears; **(2)** If the network is not identified uniquely from previous INPUT/OUTPUT pairs, generate a new random expression pattern I_1 (which corresponds to a new attractor cycle) and repeat STEP (1) and STEP (2).

The result of a computational experiment is summarized in Fig. 5. Note that, in this experiment, 10 random networks of $K = 2$ are generated for each n, and average numbers (over 10 trials) of attractors and INPUT/OUTPUT pairs, which are required to identify the network uniquely, are computed. From Fig. 5, although it is seen that the number of INPUT/OUTPUT pairs becomes much larger than one in the case of Fig. 4, the growth rates of both the number of attractors and the number of INPUT/OUTPUT pairs are still proportional to $\log n$. From Fig. 5, it is suggested under the Boolean network model of low K that, if we observe time series of gene expression patterns in several tens of different environments (or different types of cells), we can identify the network. Of course, further studies must be done in order to verify this statement, especially for real biological systems.

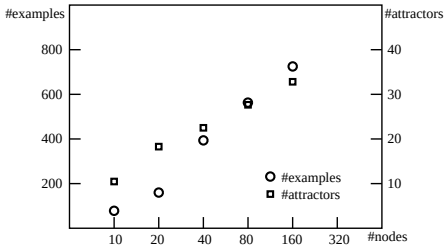


Figure 5: Results of computational experiments on the number of INPUT/OUTPUT pairs and the number of attractors required to identify the Boolean networks of $K = 2$.

5 Discussions

We have proved that $O(\log n)$ INPUT/OUTPUT pairs are necessary and sufficient for the identification of the Boolean networks of bounded indegree. For that purpose, we proposed a simple algorithm. Moreover, in oder to clarify a constant factor on $\log n$, we made computational experiments.

Of course, real biological systems are different from Boolean networks: nodes in a Boolean network take binary values which are updated synchronously, whereas quantities of gene expressions in real cells are not binary and are changing continuously in time. However, owing to its simplicity, the proposed algorithm can be extended in various way. It can be extended for networks in which the following conditions are satisfied: **(1)** There is a procedure for enumerating (possible) functions assigned to each node; **(2)** There is a procedure for testing whether or not a function assigned to each node is consistent with examples.

Moreover if the above procedures work in polynomial time, the whole algorithm also works in polynomial time. For example, we can extend the algorithm for the networks consisting of functions which depend not only on expression patterns of time t but also on expression patterns of time $t-1, t-2, \ldots, t-N$ for some constant N. We can extend for functions which take not $0/1$ but some discrete values. Mathematical analysis would also be extended for such cases.

The algorithm may also be extended for the identification of networks or systems in which continuous functions are used. As in the case of REVEAL [5], the algorithm can be extended for such systems if continuous behaviors can be approximated by discrete systems.

The drawback of the proposed algorithm is that it is not efficient: it works in $O(n^3 m)$ time even for $K = 2$, and it works in $O(n^4 m)$ time for $K = 3$.

Although the idea used in REVEAL may be useful, the efficiency of REVEAL is not enough (for higher K) as Liang *et al.* [5] pointed out. Therefore, development of faster algorithms is an important future work.

Finally, we believe that our theoretical results, along with the experimental results[5] by Liang *et al.*, encourage the attempts to discover the gene regulation mechanism from time series of gene expression patterns.

References

1. T. Akutsu, S. Kuhara, O. Maruyama and S. Miyano, Identification of gene regulatory networks by strategic gene disruptions and gene overexpressions, *Proc. 9th ACM-SIAM Symp. Discrete Algorithms*, 695(1998).
2. A. Arkin, P. Shen and J. Ross, A test case of correlation metric construction of a reaction pathway from measurements, *Science* **277**, 1275 (1997).
3. J.L. DeRisi, V.R. Lyer and P.O. Brown, Exploring the metabolic and genetic control of gene expression on a genomic scale, *Science* **278**, 680 (1997).
4. M.J. Kearns and U.V. Vazirani, *An Introduction to Computational Learning Theory*, The MIT Press (1994).
5. S. Liang, S. Fuhrman and R. Somogyi, REVEAL, a general reverse engineering algorithm for inference of genetic network architectures, *Pacific Symposium on Biocomputing* **3**, 18 (1998)
6. H.H. McAdams and L. Shapiro, Circuit simulation of genetic networks, *Science* **269**, 650 (1995).
7. R. Somogyi and C.A. Sniegoski, Modeling the complexity of genetic networks: Understanding multigene and pleiotropic regulation, *Complexity* **1**, 45 (1996).
8. R. Thomas, D. Thieffry and M. Kaufman, Dynamical behaviour of biological regulatory networks -I. Biological role of feedback loops and practical use of the concept of the loop-characteristic state, *Bulletin of Mathematical Biology* **57**, 247 (1995).
9. X. Wen *et al*, Large-scale temporal gene expression mapping of central nervous system development, *Proc. Natl. Acad. Sci. USA* **95**, 334 (1998)
10. A. Wuensche, Genomic regulation modeled as a network with basins of attraction, *Pacific Symposium on Biocomputing* **3**, 89 (1998).
11. C-H. Yuh, H. Bolouri and E.H. Davidson, Genomic Cis-regulatory logic: experimental and computational analysis of a sea urchin gene, *Science* **279**, 1896 (1998).