ISSN 1993–6842. Biopolymers and Cell. 2012. Vol. 28. N 3. P. 163–170 (Translated from Ukrainian) doi: 10.7124/bc.000036

REVIEWS

UDC 577.218

# Overview of methods of reverse engineering of gene regulatory networks: Boolean and Bayesian networks

**A. O. Frolova**

Institute of Molecular Biology and Genetics, NAS of Ukraine
150, Akademika Zabolotnogo Str., Kyiv, Ukraine, 03680

fshodan@gmail.com

*Reverse engineering of gene regulatory networks is an intensively stu- died topic in Systems Biology as it reconstructs regulatory interactions between all genes in the genome in the most complete form. The extre- me computational complexity of this problem and lack of thorough reviews on reconstruction methods of gene regulatory network is a sig- nificant obstacle to further development of this area. In this article the two most common methods for modeling gene regulatory networks are surveyed: Boolean and Bayesian networks. The mathematical descrip- tion of each method is given, as well as several algorithmic approaches to modeling gene networks using these methods; the complexity of al- gorithms and the problems that arise during its implementation are also noted.*

*Keywords: reconstruction of gene regulatory networks, Boolean networks, Bayesian networks.*

**Introduction**. The gene regulatory network is a composition of indirectly related modular DNA elements (genes), receiving multiple input signals in the form of RNA and proteins, processing the signals and inducing the rate of the network genes transcribing into RNA and translating into proteins. The network architecture reflects the interaction between its various elements and provides us with the most complete information on the regulation of cell functioning in contrast to the traditional study of single genes, thus, the reverse engineering of gene regulatory networks is an important topic of Systems Biology.

There are 10 currently used approaches to the engineering of gene networks, including machine learning, Bayesian networks, Boolean networks, differential equations, Information Theory, Petri nets, neural networks, and genetic algorithms [1–3]. Each approach has advantages and disadvantages, the definition of which is complicated due to the lack of substantial reviews in scientific literature. Another problem is that the abovementioned approaches are used to reconstruct small networks of only 10–30 genes. The increase in the number of genes causes an exponential growth of the calculation complexity: for 30 genes there are $2.71 \cdot 10^{158}$ probable network variants in case of using Bayesian networks [4], although for Information Theory the complexity estimate is considerably less [5]. However, the task of reverse-engineering of gene networks is still NP-hard [1], therefore, a vital part of reviews should be dedicated to estimating the computational complexity of the inference algorithms and the analysis of algorithms which will allow revealing the possibility of parallelizing to use them in the distributed computing and clusters.

This article reviews two engineering methods – Boolean and Bayesian networks, several algorithmic approaches and their evaluation.

Different methods of presenting gene networks. One and the same gene regulatory network may be

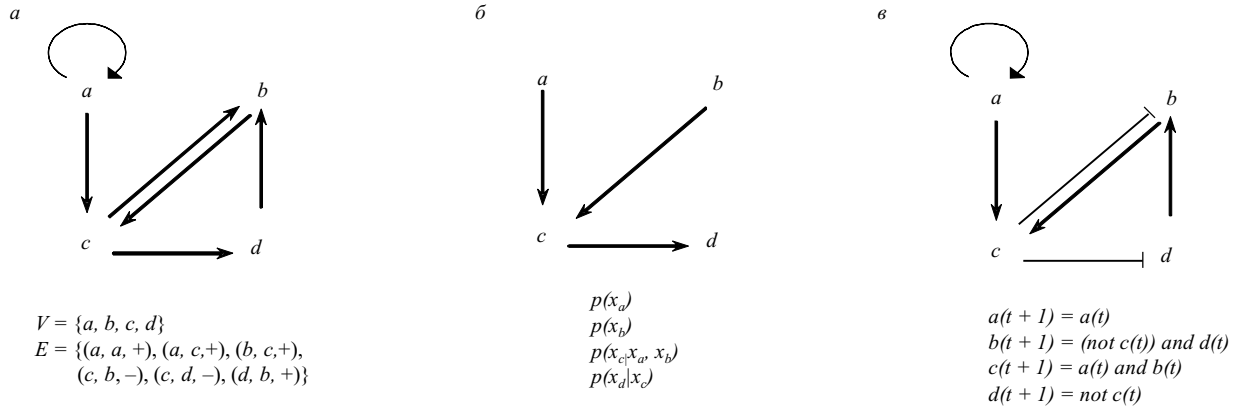


Fig. 1 Different ways of presenting the gene regulatory network of four genes *a*–*d* [6]: *a* – network in the form of the directed graph (interaction course is indicated with "+" and "–", i.e. activation and inhibition); *b* – respective model in the form of Bayesian network (it should be noted that some interactions were neglected, in particular, the inhibition of gene *b* by gene *c* and the activation of gene *b* by gene *d*, in order to obtain a network without cycles); *c* – Boolean network

represented in various ways (Fig. 1). The simplest method is directed or undirected graph.

The directed graph $G$ is a pair $<V, E>$, where $V$ – vertex set and $E$ – edge set. The vertices correspond to genes (or other system components), while the edges, indicated as a pair of vertices $<i, j>$, correspond to the regulatory interactions between the components. The graph is directed if $i$ and $j$ are the head and tail of the edge, respectively. The definitions of vertices and edges may be extended to store additional information about the genes and their interactions. For instance, an edge may be defined as $<i, j,$ *properties*$>$. A slot *properties* may indicate whether one gene inhibits (–) or activates (+) another one (see Fig. 1, *a*). Also *properties* may be the list of regulators and their effect on this edge, for instance $<i, j$ *((k, activator), (l, inhibitor as a homodimer protein)*$>$ [6].

Boolean networks. The level of gene expression in Boolean synchronous networks is defined by the binary variable which is either 0 or 1, i.e. the gene is either knocked-out or expressing. The status of genes changes at each discrete time step, that is why the networks are called synchronous. A new status of the gene may depend on the previous state of this gene and other genes. $N$ Boolean network nodes is $N$ genes of the regulatory network, $k$ inputs of each node (here $k$ is the maximal number of inputs for each node) is $k$ interactions, regulating the gene expression. $k$ inputs into a specific node determine the binary level of the expression for the corresponding gene. As each vertex may be in two states only, the network of $N$ genes has $2^N$ of different states. $N$- dimentional vector of variables may describe the state at time $t$. The value of each variable at time $t + 1$ depends on the input data, which may be computed using Boolean functions. The number of probable Boolean functions for the vertex with $k$ inputs is $2^{2k}$. Let us consider an example of the rules, used for the network, in Fig. 1, *c*:

$$a(t+1) = f_a(a(t)) = a(t), \qquad k = 1;$$
$$b(t+1) = f_b(c(t), d(t)) = (\neg c(t)) \wedge d(t), \qquad k = 2;$$
$$c(t+1) = f_c(a(t), b(t)) = a(t) \wedge b(t), \qquad k = 2;$$
$$d(t+1) = f_d(c(t)) = (\neg c(t)), \qquad k = 1.$$

These rules may be used to create the table of transitions from one state to the other, which demonstrates that this network has two types of the stationary behavior. Given the initial state $a$ equals 0, the system acquires stable state 0101, which means that genes $a$, $c$ are knock-outs, while genes $b$, $d$ are knock-ins. Given the initial state $a$ is 1, the system runs into a cyclic path, constantly running the following line of states: $1000 \rightarrow \rightarrow 1001 \rightarrow 1101 \rightarrow 1111 \rightarrow 1010 \rightarrow$ 1000 [6].

The sequence of the states, formed due to Boolean transformation, is the system trajectory. As the number of states is finite, the set of possible transitions is also finite. Thus, each trajectory leads either to the stationary state or to the cyclic state. These states are called attractors. All the states, leading to the same attractor, form the attraction basin.

*The states of Boolean network*

| $abcd \rightarrow a'b'c'd'$ | |
|---|---|
| Interpretation of states at $a = 0$ | Interpretation of states at $a = 1$ |
| $0000 \rightarrow 0001$ | $1000 \rightarrow 1001$ |
| $0001 \rightarrow 0101$ | $1001 \rightarrow 1101$ |
| $0010 \rightarrow 0000$ | $1010 \rightarrow 1000$ |
| $0011 \rightarrow 0000$ | $1011 \rightarrow 1000$ |
| $0100 \rightarrow 0001$ | $1100 \rightarrow 1011$ |
| $0101 \rightarrow 0101$ | $1101 \rightarrow 1111$ |
| $0110 \rightarrow 0000$ | $1110 \rightarrow 1010$ |
| $0111 \rightarrow 0000$ | $1111 \rightarrow 1010$ |

The Boolean networks are used to study general properties of large gene networks. Viewing the random Boolean networks (number of inputs $k$ per one gene and corresponding Boolean functions are selected at random), Kauffman [7, 8] revealed that this system demonstrates rather an ordered dynamics at small $k$ values and specific sets of rules. The average expected number of attractors is $\sqrt{N}$, and the average length of attractors is limited to the value in proportion to $\sqrt{N}$. Kauffman made an assumption about the interpretation of the number of probable attractors as the number of cells of different types. This number is in good agreement with currently known information about the types of cells [9].

The algorithm, described in [10, 11], may be used for reverse engineering of Boolean networks using the data of microarray experiments. This algorithm defines whether the vertex set $\nu_1, \nu_2, ..., \nu_k$, $k \le N$ explains the expression of a specific vertex $\nu_i$. Boolean function "activator-inhibitor", described for the vertex $\nu_i$ may be defined using the enumeration method, it is as follows

$$\nu(t) = (\nu_1(t) \vee \nu_2(t) \vee ...) \wedge \neg(\nu_j(t) \vee \nu_{j+1}(t) \vee ...),$$

where the first bracket is activator vertices, and the second one is inhibitor vertices.

Obviously, given small $k$ values, the algorithm complexity is polynomial, but it may have considerable effect on the quality of obtained network. As stated above, the number of all the possible Boolean functions equals $2^{2^k}$, thus, the increase in the value of $k$ leads to exponential complexity.

A more generalized and substantial approach to solving the problem is found in [12–14], the authors of which used the limited Boolean networks. In this case the regulatory relations are presented by the matrix $A_{n \times n}$, where $a_{ij} = 1$ at positive regulation of gene $x_i$ by gene $x_j$; $a_{ij} = -1$ at negative regulation of gene $x_i$ by gene $x_j$ and $a_{ij} = 0$ in other cases.

Thus, Boolean function $f_i$ is defined in accordance to matrix $A$ and values of genes $x_j, j = 1, ..., n$ y at time $t$:

$$x_i(t+1) = \begin{cases} 1, if \sum_j a_{ij} x_j(t) > 0; \\ 0, if \sum_j a_{ij} x_j(t) < 0; \\ x_i(t), if \sum_j a_{ij} x_j(t) = 0. \end{cases}$$

The sum $\sum a_{ij} x_j(t) > 0$ – is the input of gene $x_i$ y time $t$. As not all the Boolean functions may be defined in this representation, the Boolean network is considered to be constrained. The reconstruction of gene networks is limited to solving the constraint satisfaction problem (CSP).

CSP is defined by a set of variables $X = \{x_1, x_2, ..., x_n\}$; tuples $D = \{D_1, D_2, ..., D_n\}$, where $D_i$ – domain tuple for $x_i$; constraints $C = \{C_1, C_2, ..., C_m\}$, restricting the values, which may be accepted by the variables simultaneously, where each set of $C_i$ has constraints of the subset of variables and defines the feasible combination of values for these variables. The solution of CSP is the assigning to each variable $x_i$ the value from its domain $D_i$ to satisfy all the constraints in $C$ [15].

CSP, defined in finite domains, are usually solved by search algorithms, namely, by stepwise assignment of possible values to the variables and verification of the constraints satisfaction. The known algorithms are backtracking, constraint propagation, and local search [12]. The processes of selecting variables and assigning some values to these variables depend on the order of selection, therefore, there are many heuristic methods to solve CSP [15] which has evident effect on the reconstruction accuracy.

In addition to the abovementioned problem, there are a number of general drawbacks of the Boolean network reconstruction using the real data [6]:

1. Binarization is a complex process, significantly affecting the result. Sometimes it is hard to define the exact binarization level using the expression data.

2. The states are incomplete. In practice most transitions between the states are lost after the binarization.

3. The existence of a large number of time points is critical. Many transitions between the states are required to distinguish correct states from incorrect ones and to achieve a stable result.

4. Time points should not be very close to one another. This is an unsteady balance between the highest possible number of transitions between the states and false-positive states. If two time points are too close, the transition between them does not demonstrate any changes, as the binarization is a very rough threshold, which does not allow distinguishing insignificant concentration variations. This leads to the occurrence of a high number of false-positive cycles in the corresponding graph.

It should be noted that Boolean networks are not just a reconstruction method, rather it is the representation method, therefore, many different approaches are used for reverse engineering [16].

Thus, the Boolean gene network may be reconstructed using the Information Theory as in the known algorithm REVEAL [17]. However, the Information Theory is a specific set of the reconstruction methods requiring the discrete detailed review, so here we confine to a mere mention [18–21]. As seen, the classification of the reconstruction methods is not so strict and the complex of approaches is often used to solve the task of reverse engineering of gene networks.

**The Bayesian networks** reflect the regulatory gene networks as a directed acyclic graph $G = <V, E>$. Similarly to the definition for the usual graph, the vertices $i \in V$ correspond to the genes, and the edges – to the regulatory interactions. The variables $x_i$ belong to the vertices and define the regulatory properties, for instance, the level of gene expression or the number of active proteins. The conditional probability distribution $p(x_i \mid L(x_i))$ is defined for each $x_i$, where $L(x_i)$ – the variable, belonging to direct regulators *i.*

The directed graph *G* together with the conditional distribution describe the joint probability distribution $p(x)$, defining the Bayesian network. It may be considered as follows

$$p(x) = \prod_i p(x_i \mid L(x_i)).$$

The directed graph reflects the probability dependencies: the level of gene expression, presented by the daughter vertex, depends on the expression level of parent genes. Hence, the graph also has conditional independencies $i(x_i; y \mid z)$, which means that: $x_i$ does not depend on $y$, given the availability of $z$. Two graphs, reflecting the Bayesian network, are equivalent, if the sets of their independent relations are equal. However, in this case they may be considered only as equal undirected graphs. Completely equivalent graphs are impossible to reveal using the studies of the variable $x$ only [22].

For the network in Fig. 1, *b*, the conditional independent relations are as follows [6]:

$$i(x_a; x_b) \cdot i(x_d; x_a, x_b \mid x_c),$$

while the joint probability distribution of the network [6] is

$$p(x_a, x_b, x_c, x_d) = \\ = p(x_a) \cdot p(x_b) \cdot p(x_c \mid x_a; x_b) \cdot p(x_d \mid x_c).$$

The aim of reconstruction of gene regulatory networks from the expression data using the Bayesian networks is to find the network or the class of equivalent networks, which explain the experiment data in the best possible way. The problem is to define the initial probability distribution. However, it is more reasonable to use the dynamic Bayesian networks, which may be considered as the expansion of common Bayesian networks and which are capable of reflecting the dynamics of gene networks. Given the variable of time-series microarray experiment $x \in R^{n \times p}$, $x_{ti}$, where $n$ – the number of time points, and $p$ – the number of genes, is the observation of gene $i$ at time $t$, then the observation vector at time $t$ may be presented as $x_{(t)} = [x_{t1}, ..., x_{tp}]^T$ and $i$ gene at all the time points is $x_{(i)} = [x_{1i}, ..., x]^T$.

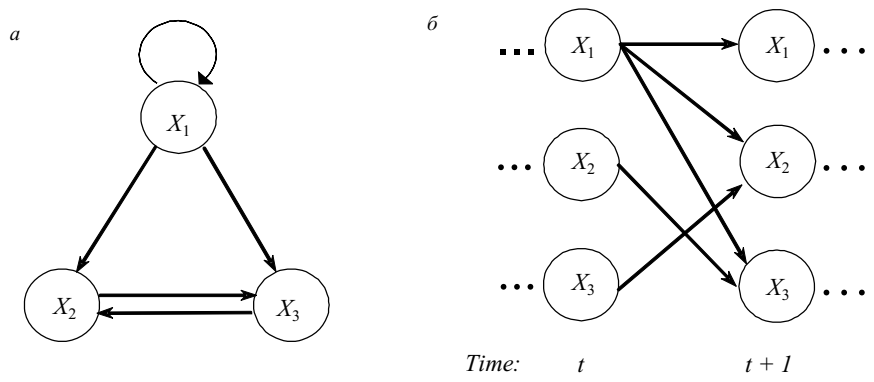


Fig. 2 The transformation of a simple network into the dynamic Bayesian network [23]: *a* – simple gene network, where $X_2$ and $X_3$ form a cycle, and $X_1$ is self-regulated; *b* – equivalent dynamic Bayesian network without cycles

The dynamic Bayesian networks assume the temporal dependence where the directed edges should "move" forward with time [23].

There is a common assumption that these networks are the first-order Markov model where each gene is directly affected only by the previous genes [24]. As these models are time-dependent, it is easy to create a network with reverse cycles. Fig. 2 demonstrates a simple transformation of a cyclic network of three genes into the acyclic dynamic Bayesian network [23].

The joint probability distribution for the dynamic Bayesian network may be calculated as

$$P(x_{11},\ldots,x_{np}) = \prod_{i=1}^{p}\prod_{t=1}^{n} P(x)_{ti} | L(x_{ti}).$$

As stated above, the main aim of the gene network inference is the creation of such networks which would present the best explanation of the experimental data. This requires to find the structure and parameters of the dynamic Bayesian network using the data. This task may be formulated as follows.

Given the data from different time points, when $D = \{x_1, ..., x_n\}$, we should find the model $M = (G, \theta)$,with the best correspondence to $D$, where $M$ is defined by the structure of the dynamic Bayesian network $G$, as well as with the corresponding parameter $\theta$ from the family of conditional probability distribution.

According to the Bayes' rule, the posterior distribution of model $M$

$$P(M|D) = \frac{P(M)P(D|M)}{P(D)},$$

where the denominator $P(D) = \sum P(D \mid M)P(M)$ – a normalizing factor, not dependent on $M$, thus, taking the logarithm, one can calculate the valuation function for $M$ [25]:

$$S(M) = \log P(D|M) + \log P(M),$$

where the parameter $P(M)$ is a priori for the model, and $P(P(D \mid M)$ – marginal probability for the $D$ data, given the model is $M$.

$P(D|M) = \int P(D|\theta, G)P(\theta|G)d\theta$, де $P(\theta|G)$ – – a priori distribution for the parameters. The selection of optimal model $M$ comes to the maximization of the marginal probability.

To evaluate the integral, one can use Dirichlet distribution for discrete polynomial distributions and Wishart distribution – for continuous gaussian distributions [23].

Even given the evaluation function, finding an optimal dynamic Bayesian network for reverse engineering of gene networks is a very complicated task. Firstly, the parent vertex set for each vertex is$2^N$, де $N$ – where $N$ is the total number of nodes. Therefore, the optimization task of finding the model with the highest evaluation function is exponential [26].

Secondly, the search algorithm is not always successful in finding the best model, usually only a local maximum is reached, therefore, the only selected model with the maximal valuation function is not always the best.

There are several traditional approaches to solving the above task. One of them is a greedy hill-climbing search with random restarts [27]. A random network model is selected for each restart. The mutation of this basic structure occurs via addition or subtraction of one edge. The algorithm defines all the possible mutations of the basic structure and selects the one with the

highest score, and then it becomes the basic one. This procedure is repeated until the local maximum is reached, then the model is saved, and the procedure restarts. The result is the set of models, the number of which equals to the number of restarts. This algorithm is defined in the work [23]; it is formulated in pseudo-code as follows:

**Greedy Hill-Climbing Search with Restarts for DBN**

**Input**: $D$ (test data of time points))
$N_{res}$ (number of restarts)
**Output**: $M_{out}$ (set of models with the highest grades)

**for** $i = 1$ *to* $N_{res}$ **do**
*produce random structure* $M_0$
**repeat**
$M_{best} \leftarrow M_0$
**foreach** *pair of nodes in DBN* **do**
**if** *edge* = 0 (no connection between two vertices)
**then**
$M' \leftarrow addEdge(M_0)$
**else**
$M' \leftarrow removeEdge(M_0)$
**end**
**if** $score(M') > Score(M_{best})$ **then**
$M_{best} \leftarrow M'$
**end**
**end**
**until** $M_{best} = M_0$ (local maximum is reached)
**return** $M_{out} \leftarrow M_{best}$
**end**.

Another class of heuristic algorithms, used to solve the task of selecting the best model is theMarkov Chain Monte Carlo (MCMC) method [28] with the multivariate complex distribution. The mechanism of this method is the creation of Markov chain, where a new model $\widetilde{M}$ is generated only based on the previous $M$. Finally there is a chain of models, coinciding with the expected distribution. A sufficient condition for the coincidence is the balance equation for all the models [23]:

$$P(M_i|M_k)P(M_k|D) = P(M_k|M_i)P(M_i|D),$$

where $P(M_i|M_k)$ – transition probability from $P(M_k)$ to $P(M_i)$.

One of important MCMC algorithms is Metropolis-Hastings algorithm, based on the algorithm of sampling – acceptance-rejection sampling algorithm [29]. At each restart the algorithm generates a new model, the candidate distribution $Q(\widetilde{M}, M)$, which is the probability of return of a new model $\widetilde{M}$ **at** given model $M$. Given the candidate model $\widetilde{M}$, it is possible to calculate the probability of its acceptance

$$\alpha(\widetilde{M}, M) = \min\left\{1, \frac{P(\widetilde{M}|D)Q(M|\widetilde{M})}{P(M|D)Q(\widetilde{M}|M)}\right\},$$

If the probability satisfies these conditions, the Markov chain selects the current candidate model. Let us refer to the work [23] once again to illustrate the algorithm in pseudocode:

**Metropolis-Hastings sampling algorithm for DBN**

**Input**: $D$ (test data of time points)
$N_{sam}$ (number of samples)
**Output**: $M_{out}$ (chain of models)

Produce initial model $M_0$
**for** $i = 1$ *to* $N_{sam}$ **do**
sample a new model $\widetilde{M}$ from $Q(\widetilde{M}, M)$
compute
$\alpha(\widetilde{M}, M) = \min\left\{1, \frac{P(\widetilde{M}|D)Q(M_i|\widetilde{M})}{P(M_i|D)Q(\widetilde{M}|M_i)}\right\},$
sample $u$ from $U_{(0,1)}$ (uniform distribution *to* (0,1))
**if** $\alpha(\widetilde{M}, M_i) > u$ **then**
$M_i + 1 \leftarrow \widetilde{M}$
**else**
$M_i + 1 \leftarrow M_i$
**end**
**return** $M_{out} \leftarrow M_{i+1}$
**end**.

Besides, the application of MCMC to find the optimal Bayesian network is a costly computational task. An increase in the number of network nodes results in the exponential increase in the algorithm complexity [30]. Compared to the greedy hill-climbing search with random restarts, MCMC demonstrates better results and is faster [23].

One of the examples of Bayesian networks inference software is Banjo, which studies the structure of static and dynamic Bayesian networks (http://www.cs.duke.edu/~amink/software/banjo/). The authors used the greedy hill-climbing search with random restarts, simulated annealing and genetic algorithms, which demonstrated similar results under the condition of long-term performance. However, each method requires different different time to find the best network (20 genes per 2,000 points): on Dell PC, 2.26 GHz CPU, 1 GB RAM the greedy search was the fastest (minutes), simulated annealing rated (dozens of minutes), while the genetic algorithm was the slowest (hours) 31].

The Bayesian networks may be efficiently combined with other methods. For instance, in [32] the greedy hill-climbing search by Banjo realization was used along with LASSO (least absolute shrinkage and selection operator) and Dantzig selector from the fa mily of regressive methods.

**Conclusions**. The analysis of two different approaches to solving the problem of reverse engineering of gene networks demonstrates that the increase in the number of genes in the network leads to the exponentially complicated computational task.

In case of Boolean networks the binarization of the gene expression value (active or passive) allows studying larger networks investigating their general properties. Besides, we can limit the number of regulators of a specific gene, thus facilitating the algorithm to save some time. However, the abovementioned facilitations affect the network quality.

In case of Bayesian networks the presented algorithms demonstrate that there is a high risk of getting into a local maximum, as the exponential complexity requires the application of heuristic algorithms.

In the author's opinion better results in both cases can be obtained by distributing the computing load using the cluster of computers. It does not require the algorithm parallelism, a simpler way is to distribute the data among the cluster nodes. In addition, it is reasonable to use ensemble-methods, i.e. combinations of several approaches, which will definitely enhance the engineering accuracy.

Thus, one may conclude that the algorithms of reverse engineering of gene networks on the basis of Boolean and Bayesian networks require detailed mathematical foundation, which would allow reconstructing the network model with the most accurate correspondence to the experimental data, as well as the modern computing approaches in the informational technology fields, sincethe considered methods do not solve the problem of exponential search.

The author would like to express her gratitude to Professor M. Yu. Obolenska, Dr. Sci. (biology), and B. T. Tokovenko, Ph. D. (biology), (the Institute of Molecular Biology and Genetics, NAS of Ukraine) for their valuable advice and relevant remarks.

*А. О. Фролова*

Огляд методів моделювання мереж
генної регуляції: булеві і баєсові мережі

Інституту молекулярної біології і генетики НАН України
Вул. Академіка Заболотного, 150, Київ, Україна, 03680

Резюме

*Однією з проблем сучасної системної біології є моделювання мереж генної регуляції, які у найповнішому вигляді відтворюють регуляторні взаємодії між генами всього організму. Надзвичайна обчислювальна складність цієї задачі та відсутність ґрунтовних оглядів методів реконструкції генних мереж є значною перешкодою для подальшого розвитку цього напрямку системної біології. У даній статті розглянуто два найпоширеніших методи моделювання мереж генної регуляції: булеві і баєсові мережі, та наведено математичний опис кожного з них, а також розкрито декілька алгоритмічних підходів до моделювання генних мереж за допомогою цих методів, вказано на складність алгоритмів та зазначено проблеми, що виникають при їхньому застосуванні.*

*Ключові слова: реконструкція мереж генної регуляції, булеві мережі, баєсові мережі.*

*А. О. Фролова*

Обзор методов моделирования сетей генной регуляции:
булевы и баесовы сети

Резюме

*Одна из проблем современной системной биологии – моделирование сетей генной регуляции, в наиболее полной мере отображающих регуляторные взаимодействия между генами всего организма. Большая вычислительная сложность такой задачи и отсутствие основательных обзоров методов реконструкции генных сетей являются значительной преградой для дальнейшего развития этого направления системной биологии. В данной статье рассмотрены два наиболее распространенных метода моделирования сетей генной регуляции: булевые и баесовые сети, а также дано их математическое описание, а также раскрыто несколько алгоритмических подходов к моделированию генных сетей с помощью этих методов, указаны сложность алгоритмов и проблемы, которые возникают при их использовании.*

*Ключевые слова: реконструкция сетей генной регуляции, булевые сети, баесовые сети.*

REFERENCES


1. *Lee W.-P., Tzou W.-S.* Computational methods for discovering gene networks from expression data // Brief. Bioinform.–2009.–**10**, N 4.–P. 408–423.
2. *Hecker M., Lambeck S., Toepfer S., van Someren E., Guthke R.* Gene regulatory network inference: data integration in dynamic models – a review // Biosystems.–2009.–**96**, N 1.–P. 86–103.
3. *Karlebach G., Shamir R.* Modelling and analysis of gene regulatory networks // Nat. Rev. Mol. Cell Biol.–2008.–**9**.–P. 770–780.
4. *Ott S., Imoto S., Miyano S.* Finding optimal models for small gene networks // Pac. Symp. Biocomp.–2004.–**9**.–P. 557–567.
5. *Margolin A. A., Nemenman I., Basso K., Wiggins C., Stolovitzky G., Favera R. D., Califano A.* ARACNE: An algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context // BMC Bioinformatics.–2006.–**7**, Suppl. 1.–S 7.
6. *Klipp E.* Systems biology in practice: concepts, implementation and application.–New York: Wiley-VCH, 2005.–465 p.
7. *Kauffman S.* Antichaos and adaptation // Sci. Am.–1991.–**265**, N 2.–P. 78–84.
8. *Kauffman S.* The Origins of Order.–Oxford: Univ. press, 1993.–709 p.
9. *Kauffman S.* Investigations.–Oxford: Univ. press, 2002.–308 p.
10. *Akutsu T., Miyano S., Kuhara S.* Identification of genetic networks from a small number of gene expression patterns under the Boolean network model // Pac. Symp. Biocomp.–1999.–**4**.–P. 17–28.
11. *Martin S., Zhang Z., Martino A., Faulon J.* L. Boolean dynamics of genetic regulatory networks inferred from microarray time series data // Bioinformatics.–2007.–**23**, N 7.–P. 866.
12. *Higa C., Louzada V., Andrade T., Hashimoto R.* Constraint-based analysis of gene interactions using restricted boolean networks and time-series data // BMC Proceedings.–2011.–**5**, Suppl. 2.–S 5.
13. *Lau K., Ganguli S., Tang C.* Function constrains network architecture and dynamics: a case study on the yeast cell cycle Boolean network // Phys. Rev. E. Stat. Nonlin. Soft Matter Phys.–2006.–**75**, N 5, pt 1.–051907.
14. *Xia Q., Liu L., Ye W., Hu G.* Inference of gene regulatory networks with the strong-inhibition Boolean model // New J. Phys.–2011.–**13**, N 8.–083002.
15. *Tsang E. P. K.* Foundations of constraint satisfaction.–London; San Diego: Acad. press, 1993.–405 p.
16. *Lee W.-P., Tzou W.-S.* Computational methods for discovering gene networks from expression data // Brief Bioinform.–2009.–**10**, N 4.–P. 408–423.
17. *Liang S., Fuhrman S., Somogyi R.* Reveal, a general reverse engineering algorithm for inference of genetic network architectures // Pac. Symp. Biocomp.–1998.–**3**.–P. 22.
18. *Zola J., Aluru M., Aluru S.* Parallel information theory based construction of gene regulatory networks // Hipc.–2008.–**5374**.–P. 336–349.
19. *Margolin A. A., Nemenman I., Basso K., Wiggins C., Stolovitzky G., Favera R. D., Califano A.* ARACNE: An algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context // BMC Bioinformatics.–2006.–**7**, suppl. 1.–S 7.
20. *Zola J., Aluru M., Sarje A., Aluru S.* Parallel information-theory-based construction of genome-wide gene regulatory networks // IEEE Transactions on Parallel and Distributed Systems.–2010.–**21**, N 12.–P. 1721–1733.
21. *Daub C. O., Steuer R., Selbig J., Kloska S.* Estimating mutual information using B-spline functions-an improved similarity measure for analyzing gene expression data // BMC Bioinformatics.–2004.–**5**–P. 118.
22. *Friedman N., Linial M., Nachman I., Pe'er D.* Using Bayesian networks to analyze expression data // J. Comp. Biol.–2000.–**7**, N 3–4.–P. 601–620.
23. *Wu H., Liu X.* Dynamic bayesian networks modeling for inferring genetic regulatory networks by search strategy: Comparison between greedy hill climbing and mcmc methods // Proc. World Acad. Sci., Engin. Technol.–2008.–**34**.–P. 224–234.
24. *Sima C., Hua J., S. Jung S.* Inference of gene regulatory networks using time-series data: A survey // Curr. Genomics.–2009.–**10**, N 6.–P. 416–429.
25. *Yu J., Smith V. A., Wang P. P., Hartemink A. J., Jarvis E. D.* Using Bayesian network inference algorithms to recover molecular genetic regulatory networks // 3rd Int. Conf. Syst. Biol. (ICSB02).–Stockholm, 2002.
26. *Chickering D., Heckerman D., Meek C.* Large-sample learning of Bayesian networks is NP-hard // J. Mach. Learn. Res.–2004.–**5**.–P. 1287–1330.
27. *De Campos L., Fernandez-Luna J., Puerta J.* An iterated local search algorithm for learning Bayesian networks with restarts based on conditional independence tests // Int. J. Intellig. Syst.–2003.–**18**, N 2.–P. 221–235.
28. *Scollnik D.* An introduction to Markov Chain Monte Carlo methods and their actuarial applications // Proc. Casualty Actuarial Soc.–1996.–**83**.–P . 114–165.
29. *Chib S., Greenberg E.* Understanding the Metropolis-Hastings algorithm // Am. Statistic.–1995.–**49**, N 4.–P. 327–335.
30. *Friedman N., Koller D.* Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks // Machine Learning.–2003.–**50**, N 1.–P. 95–125.
31. *Yu J., Smith V., Wang P., Hartemink,A., Jarvis E.* Advances to Bayesian network inference for generating causal networks from observational biological data // Bioinformatics.–2004.–**20**, N 18.–P. 3594–3603.
32. *Vignes M., Vandel J., Allouche D., Ramadan-Alban N., Cierco-Ayrolles C., Schiexet T., Mangin B., de Givry B.* Gene regulatory network reconstruction using Bayesian networks, the dantzig selector, the lasso and their meta-analysis // PLoS ONE.–2011.–**6**, N 12.–e29165.


Received 11.11.11