FINDING OPTIMAL MODELS FOR SMALL GENE NETWORKS

S. OTT, S. IMOTO, S. MIYANO

Human Genome Center, Institute of Medical Science, The University of Tokyo
4-6-1 Shirokanedai, Minato-ku, Tokyo 108-8639, Japan
{*ott,imoto,miyano*}*@ims.u-tokyo.ac.jp*

Finding gene networks from microarray data has been one focus of research in recent years. Given search spaces of super-exponential size, researchers have been applying heuristic approaches like greedy algorithms or simulated annealing to infer such networks. However, the accuracy of heuristics is uncertain, which - in combination with the high measurement noise of microarrays - makes it very difficult to draw conclusions from networks estimated by heuristics. We present a method that finds optimal Bayesian networks of considerable size and show first results of the application to yeast data. Having removed the uncertainty due to the heuristic methods, it becomes possible to evaluate the power of different statistical models to find biologically accurate networks.

1 Introduction

Inference of gene networks from gene expression measurements is a major challenge in Systems Biology. If gene networks can be infered correctly, it can lead to a better understanding of cellular processes, and, therefore, have applications to drug discovery, disease studies, and other areas.

Bayesian networks are a widely used approach to model gene networks[3,4,7,9,11,13,14,17]. In Bayesian networks, the behaviour of the gene network is modeled as a joint probability distribution for all genes. This allows a very general modeling of gene interactions. The joint probability distribution can be decomposed as a product of conditional probabilities $P(X_g|X_1,\ldots,X_n)$, representing the regulation of a gene g by some genes $g_1,\ldots,g_n$. This decomposition can be represented as a directed acyclic graph. The Bayesian network model has been shown to allow finding biologically plausible gene networks[4,9].

However, the difficulty of learning Bayesian networks lies in its large search space. The search space for a gene network of n genes is the space of directed acyclic graphs with n vertices. A recursive formula as well as an asymptotic expression for the number of directed acyclic graphs with n vertices (c_n) was derived by Robinson[15]. We state the asymptotic expression here:

$$c_n = \frac{n! \cdot 2^{\frac{n}{2}\cdot(n-1)}}{r \cdot z^n}; r \sim 0.57436; z \sim 1.4881 \quad (1)$$

For example, there are roughly $2.34 \cdot 10^{72}$ possible networks with 20 genes, and about $2.71 \cdot 10^{158}$ possible solutions for a gene network with 30 genes. Even for a gene network of 9 genes (search space size roughly $1.21 \cdot 10^{15}$), a brute force approach would take years of computation time even on a supercomputer. Moreover, it is known that the problem of finding an optimal network is NP-hard[1], even for the discrete scores BDe[2,3] and MDL[3]. Therefore, researchers have so far used heuristic approaches like simulated annealing[8] or greedy algorithms[9] to estimate Bayesian networks[18].

However, since the accuracy of heuristics is uncertain, it is difficult to base conclusions on heuristically estimated networks. In order to overcome this problem, we have analysed the structure of the super-exponential search space and developed an algorithm that finds the optimal solution within the super-exponential search space in exponential time. This approach is feasible for gene networks of 20 or more genes, depending on the concrete probability distribution used. Furthermore, adding biologically justified assumptions, the optimal network can be infered for gene networks of up to 40 genes.

Overcoming the uncertainties of heuristics opens up the possibility to compare statistical models with respect to their power to infer biologically accurate gene networks. Also, this method is a valuable tool for refining gene networks of known functional groups of genes.

We present the method in Section 2. In Section 3, we present results of an application of this method, which show that it can estimate gene networks biologically accurate.

2 The Method

2.1 Preliminaries

Throughout this section, we assume we are given a set of genes G and a network score function as used by several groups[4,9,17], i.e. a function $s : G \times 2^G \rightarrow \mathbb{R}$ that assigns a score to a gene $g \in G$ and a set of parent genes $A \subseteq G$. Given a network N, the score of N is defined as $score(N) =_{def} \sum_{g \in G} s(g, P^N(g))$, where $P^N(g)$ denotes the set of g's parents in N.

Examples:

1. BDe score[2,3]
   The score is proportional to the posterior probability of the network, given the data. When the BDe score is used, the microarray data needs to be discretized.

2. MDL score[3]
   The MDL score makes use of the minimal description length principle and also uses discretized data.

3. BNRC score[9]
   The BNRC score uses nonparametric regression to capture nonlinear gene interactions. Since the data does not need to be discretized, no information is lost.

The task of infering a network is to find a set of parent genes for each gene, such that the resulting network is acyclic and the score of the network is minimal.
We introduce some notations needed to describe the algorithm.

Definition 1: F
We define $F : G \times 2^G \to I\!R$ as $F(g, A) =_{def} \min_{B \subseteq A} s(g, B)$ for all $g \in G$ and $A \subseteq G$. •

The meaning of $F(g, A)$ is, by the definition, the optimal choice of parents for gene g, when parents have to be selected from the subset A. For every acyclic graph, there is an ordering of the vertices, such that all edges are oriented in the direction of the ordering. Conversely, when given a fixed order of G, we can think of the set of all graphs that comply with the given order, as we do in the next definition.

An ordering of a set $A \subseteq G$ can be described as a permutation $\pi : \{1, \ldots, |A|\} \to A$. Let us use Π^A to denote the set of all permutations of A.

Definition 2: π-linearity
Let $A \subseteq G$ and $\pi \in \Pi^A$. Let $N \subseteq A \times A$ be a network. We say N is *π-linear* iff for all $(g, h) \in N$ $\pi^{-1}(g) < \pi^{-1}(h)$ holds. •

Now we use the above definitions and define function Q^A, which will allow us to compute the score of the best π-linear network for a given π, as we show below.

Definition 3: Q^A
Let $A \subseteq G$. We define $Q^A : \Pi^A \to I\!R$ as

$$Q^A(\pi) =_{def} \sum_{g \in A} F(g, \{h \in A | \pi^{-1}(h) < \pi^{-1}(g)\}). \tag{2}$$

for all $\pi \in \Pi^A$. •

If we can compute the best π-linear network for a given permutation π using functions F and Q, then what we need to do in order to find the optimal network is to find the optimal permutation π, which yields the global minimum. Formally, we define function M for this step.

Definition 4: M
We define $M : 2^G \to \bigcup_{A \subseteq G} \Pi^A$ as

$$M(A) =_{def} \arg \min_{\pi \in \Pi^A} Q^A(\pi) \tag{3}$$

for all $A \subseteq G$. •

2.2 The Algorithm

Using above notations, the algorithm can be defined as follows.

Step 1:	Compute $F(g, \emptyset) = s(g, \emptyset)$ for all $g \in G$.
Step 2:	For all $A \subseteq G$, $A \neq \emptyset$ and all $g \in G$, compute $F(g, A)$ as $\min\{s(g, A), \min_{a \in A} F(g, A - \{a\})\}$.
Step 3:	Set $M(\emptyset) = \emptyset$.
Step 4:	For all $A \subseteq G$, $A \neq \emptyset$, do the following two steps:
Step 4a:	Compute $g^* = \arg\min_{g \in A}(F(g, A - \{g\}) + Q^{A-\{g\}}(M(A - \{g\})))$.
Step 4b:	For all $1 \leq i < \|A\|$, set $M(A)(i) = M(A - \{g^*\})(i)$, and $M(A)(\|A\|) = g^*$.
Step 5:	return $Q^G(M(G))$.

In the recursive formulas given in Step 2 and in Step 4, we want to compute the function F resp. M for a subset $A \subseteq G$ of cardinality $m = |A|$, and need function values of function F resp. M for subsets of cardinality $m - 1$. Therefore, we can apply dynamic programming in Step 2 as well as in Step 4 to compute functions F resp. M for subsets A of increasing cardinality. In the recursive formula in Step 4, first the last element g^* of the permutation $M(A)$ is computed in Step 4a, and then $M(A)$ is set in Step 4b.

2.3 Correctness and Time Complexity

First, we prove the correctness of the algorithm. The correctness of the recursive formula in Step 2 of the algorithm follows directly from the definition of F. Therefore, after execution of Step 1 and Step 2, the values of function

F for all genes g and all subsets $A \subseteq G$ are stored in the memory. Before proceeding to Step 3 and Step 4, we state a lemma on the meaning of function Q^A.

Lemma 1

Let $A \subseteq G$ and $\pi \in \Pi^A$. Let $N^* \subseteq A \times A$ be a π-linear network with minimal score. Then, $Q^A(\pi) = score(N^*)$ holds.

Proof. In a π-linear graph, a gene g can only have parents h, which are upstream in the order coded by π, that is, $\pi^{-1}(h) < \pi^{-1}(g)$. Therefore, when selecting parents for g, we are restricted to $B = \{h \in A | \pi^{-1}(h) < \pi^{-1}(g)\}$, and $F(g, B)$ is the optimal choice in this case. Since in a π-linear graph, all edges comply with the order coded by π, we can choose parents in this way for all genes independently, which proves the claim. $\triangle$

Using Lemma 1, we prove that function M can be computed by the formula given in Step 4.

Lemma 2

Let $A \subseteq G$. Let $g^* = \arg\min_{g \in A}(F(g, A - \{g\}) + Q^{A-\{g\}}(M(A - \{g\})))$. Define $\pi \in \Pi^A$ by $\pi(i) = M(A - \{g^*\})(i)$, and $\pi(|A|) = g^*$. Then, $\pi = M(A)$.

Proof. Let $\pi' \in \Pi^A$. By the definition of M, we have to show $Q^A(\pi) \leq Q^A(\pi')$. Let N^* be an optimal π-linear network, M^* be an optimal π'-linear network. Then, by Lemma 1, $Q^A(\pi) \leq Q^A(\pi')$ is equivalent to $score(N^*) \leq score(M^*)$. Let us denote the last element of π' as $h = \pi'(|A|)$. We note that for any $B \subseteq G$, $Q^B(M(B))$ is the score of a global optimal network on B by above definitions. Therefore, we have:

$$\begin{aligned} score(M^*) &= s(h, P^{M^*}(h)) + \textstyle\sum_{g \in A - \{h\}} s(g, P^{M^*}(g)) \\ &\geq s(h, P^{M^*}(h)) + Q^{A-\{h\}}(M(A - \{h\})) \\ &\geq F(h, A - \{h\}) + Q^{A-\{h\}}(M(A - \{h\})) \\ &\geq \min_{h \in A}(F(h, A - \{h\}) + Q^{A-\{h\}}(M(A - \{h\}))) \\ &= F(g^*, A - \{g^*\}) + Q^{A-\{g^*\}}(M(A - \{g^*\})) \\ &= score(N^*), \end{aligned}$$

which shows the claim. $\triangle$

Since Q can be directly computed using F, the algorithm can compute

$Q^G(M(G))$ in Step 5. Finally, $Q^G(M(G))$ is the score of an optimal Bayesian network by definition, which shows the correctness.

If the information of the best parents is stored together with $F(g, A)$ for every gene g and every subset $A \subseteq G$, the optimal network can be constructed during the computation of $Q^G(M(G))$.

Theorem 1

Optimal networks can be found using $O(n \cdot 2^n)$ dynamic programming steps.

Proof. The dynamic programming in Step 1 and Step 2 requires $O(n \cdot 2^n)$ ($n = |G|$) steps and in each step one score is computed. In the dynamic programming in Step 3 and Step 4 $O(2^n)$ steps are needed, where each steps involves looking up some previously stored scores. Note that the function Q^A does not need to be actually computed in Step 4a, because $Q^{A-\{g\}}$ can be stored together with $M(A - \{g\})$ in previous steps.
Therefore, the overall time complexity is $O(n \cdot 2^n)$. △

In biological reality, while the number of children of a regulatory gene may be very high, the number of parents can be assumed to be limited. When we limit the number of parents, the number of score calculations reduces substantially, allowing the computation of larger networks.
We state the following trivial corollary, which is practically very meaningful (see Section 3).

Corollary 1

Let $m \in \mathbb{N}$ be a constant. Optimal networks, in which no gene has more than m parents, can be found in $O(n \cdot 2^n)$ dynamic programming steps.

If we do not want to limit the number of parents by a constant, but instead can select for each gene a fixed number of candidate parents, the complexity changes as follows.

Corollary 2

Let $m \in \mathbb{N}$ be a constant. For each $g \in G$, let $C_g \subseteq G$ be a set with $|C_g| \leq m$. Optimal networks, in which each gene g has parents only in C_g, can be found in $O(2^n)$ dynamic programming steps.

Proof. Since the parents of each gene are selected from a set of constant size, the complexity of the dynamic programming in Step 1 and Step 2 becomes

constant. Therefore, the overall complexity becomes $O(2^n)$. $\triangle$

We note, that the two applications of dynamic programming in our algorithm can be implemented as a single application of dynamic programming, because when we compute function M for a set of size m, we only need function values of function F for a set of size $m-1$. Therefore, only the function values for functions F and M for sets of size $m-1$ and m need to be stored in the memory at the same time. This is practically meaningful to reduce the required amount of memory.

We also note that the algorithm can be modified to also compute suboptimal solutions. Computing the second-best or the third-best network might be valuable in order to assess the stability of the infered networks under marginal changes of the score.

3 Results

The algorithm described above was implemented as a C++ program. As scoring functions, existing implementations of the BNRC score, the BDe score and the MDL score are used. All three approaches (Theorem 1, Corollary 1 and 2) were implemented.

We applied the program to a dataset of 173 microarrays, measuring the response of *Saccharomyces cerevisiae* to various stress conditions.[5]

3.1 Application to Heat Shock Data

From the dataset we selected 15 microarrays from $25°C$ to $37°C$ heat shock experiments and 5 microarrays from heat shock experiments from various temperatures to $37°C$. Then we selected a set of 9 genes, which are involved or putatively involved in the heat shock response. Figure 1 shows the optimal network with respect to the BNRC score.

We observe that the transcription factor *MCM1* is estimated to regulate three other genes, while it is not regulated by one of the genes in this set, which is plausible. The second transcription factor in our set of genes, *HSF1*, is estimated to regulate three other heat shock genes. It is also estimated to be regulated by a *HSP70*-protein (*SSA1*), which was reported before[16]. Another chaperone among these genes, *SSA3*, also seems to play an active role in the heat shock response and interacts with *SSA1* and *HSP104*, coinciding with a report by Glover and Lindquist[6].

Overall, the result is biologically plausible and gives an indication for the active role of the chaperones *SSA1* and *SSA3* during the heat shock response.

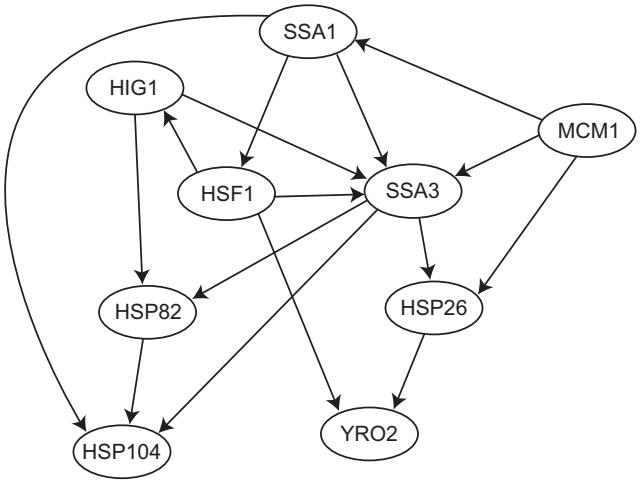

Figure 1. A gene network infered by our method.

We conclude that optimally infered gene networks are meaningful and useful for the elucidation of gene regulation.

gene	**annotation**
HSF1	heat shock transcription factor
SSA1	ER and mitochondrial translocation, cytosolic HSP70
SSA3	ER and mitochondrial translocation, cytosolic HSP70
HIG1	heat shock response, heat-induced protein
HSP104	heat shock response, thermotolerance heat shock protein
MCM1	transcription, multifunctional regulator
HSP82	protein folding, HSP90 homolog
YRO2	unknown, putative heat shock protein
HSP26	diauxic shift, stress-induced protein

3.2 Computational Possibilities and Limitations

While even networks of small scale like the network infered in Section 3.1 cannot be infered with a brute force approach (Eqn. 1), they can be optimally infered by our program using a single Pentium CPU with 1.9 GHz for about 10 minutes. In order to evaluate the practical possibilities of this approach, we selected 20 genes with known active role in gene regulation[12] from the

data set and estimated a network with optimal BNRC score using all 173 microarrays. The computation finished within about 50 hours using a Sun Fire 15K supercomputer with 96 CPUs, 900MHz each. As a result of this computational experiment, we conclude that our method is feasible for gene networks of 20 genes, even if no constraints are made and a complex scoring scheme like the BNRC score is used. For the discrete scores BDe and MDL, which can be computed much faster, even networks of more than 20 genes can be infered optimally without constraints.

When the number of parents is limited to about 6 (Corollary 1) or, alternatively, sets of about 20 candidate parents are preselected (Corollary 2), even with the BNRC score gene networks of more than 30 genes can be infered optimally. However, the method as it is now will not allow to estimate networks of more than about 40 genes.

While the theoretical time complexity of the approach given in Corollary 2 is below the time complexity of the approach given in Corollary 1, we argue that the latter might be practically more important. First, limiting the number of parents by a constant can be easily done and is biologically justified, while selecting a set of candidate parents for each gene requires a method of gene selection, which can potentially bias the computation result. Second, it has to be considered that each dynamic programming step in the computation of function F requires the computation of one score, while one dynamic programming step for function M only requires looking up some previous results. When the number of parents is limited as in Corollary 1, the required number of score calculations becomes a polynomial, which makes this approach faster in practical applications, though the approach in Corollary 2 is theoretically superior.

4 Conclusion

We have presented a method that allows to infer gene networks of 20-40 genes optimally, depending on the probability distribution used and on whether additional assumptions are made or not. This makes it possible to compare different scoring schemes, to assess the best parameters for a given scoring scheme, and to evaluate the usefulness of given microarray data, since optimal solutions are obtained. Also, the method is especially useful in settings where researchers focus on a certain group of genes and want to exploit gene expression measurements concerning these genes to the full extent.

In contrast to heuristic approaches, if the results are unsatisfying or contradictory to biological knowledge, it can be concluded that the statistical model is incorrect or the data is insufficient. Even for a network of 20 genes,

getting to know the best network from the huge search space given is a large amount of information.

We note that the method is not dependent on a certain scoring scheme or a certain kind of gene expression measurements. It can be applied in any setting, where a score as defined in Section 2 is given. For example, when sequence information[19], protein interaction data[10], or other knowledge is incorporated in the score function, this method can also be applied.

In order to find gene networks with more than 40 genes, two directions of future work open up. First, if a part of the set of subsets, in which the algorithm performs the actual search, can be pruned, the limit of feasibility might be increased. Second, compartmentalization of gene networks[18] might be used to decompose larger networks in smaller parts, and infer each partial network optimally.

Acknowledgements

The authors would like to thank Michiel de Hoon for discussions of the manuscript, and Hideo Bannai for advice on implementational issues.

References

1. D.M. Chickering. Learning Bayesian networks is NP-complete. In D. Fisher and H.-J. Lenz, editors, *Learning from Data: Artificial Intelligence and Statistics V*, Springer-Verlag, 1996.
2. G.F. Cooper, E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9: 309–347, 1992.
3. N. Friedman, M. Goldszmidt. Learning Bayesian networks with local structure. Jordan, M.I. (ed.), Kluwer Academic Publishers, pp. 421–459, 1998.
4. N. Friedman, M. Linial, I. Nachman, D. Pe'er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7: 601–620, 2000.
5. A.P. Gasch, et. al. Genomic expression programs in the response of yeast cells to environmental changes. *Molecular Biology of the Cell*, 11: 4241–4257, 2000.
6. J.R. Glover, S. Lindquist. Hsp104, Hsp70, and Hsp40: a novel chaperone system that rescues previously aggregated proteins. *Cell*, 94: 73–82, 1998.
7. A.J. Hartemink, D.K. Gifford, T.S. Jaakkola, R.A. Young. Using graphical models and genomic expression data to statistically validate models

of genetic regulatory networks. *Pacific Symposium on Biocomputing*, 6: 422–433, 2001.

8. A.J. Hartemink, D.K. Gifford, T.S. Jaakkola, R.A. Young. Combining location and expression data for principled discovery of genetic regulatory network models. *Pacific Symposium on Biocomputing*, 7: 437–449, 2002.
9. S. Imoto, T. Goto, S. Miyano. Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. *Pacific Symposium on Biocomputing*, 7: 175–186, 2002.
10. T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, Y. Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proceedings of the National Academy of Sciences USA*, 97: 4569–4574, 2001.
11. S. Imoto, S. Kim, T. Goto, S. Aburatani, K. Tashiro, S. Kuhara, S. Miyano. Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network. *Journal of Bioinformatics and Computational Biology*, in press, 2003.
12. T.I. Lee, N.J. Rinaldi, F. Robert, et. al. Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, 298: 799–804, 2002.
13. I.M. Ong, J.D. Glasner, D. Page. Modelling regulatory pathways in *E. coli* from time series expression profiles. *Bioinformatics*, 18: 241–248, 2002.
14. D. Pe'er, A. Regev, G. Elidan, N. Friedman. Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 17: 215–224, 2001.
15. R.W. Robinson. Counting labeled acyclic digraphs. *New Directions in the Theory of Graphs*, pp. 239–273, 1973.
16. Y. Shi, D.D. Mosser, R.I. Morimoto. Molecular chaperones as HSF1-specific transcriptional repressors. *Genes&Development*, 12: 654–666, 1998.
17. V.A. Smith, E.D. Jarvis, A.J. Hartemink. Evaluating functional network inference using simulations of complex biological systems. *Bioinformatics*, 18: 216–224, 2002.
18. E.P. van Someren, L.F.A. Wessels, E. Backer, M.J.T. Reinders. Genetic network modeling. *Pharmacogenomics*, 3(4): 507–525, 2002.
19. Y. Tamada, S. Kim, H. Bannai, S. Imoto, K. Tashiro, S. Kuhara, S. Miyano. Estimating gene networks from gene expression data by combining Bayesian network model with promoter element detection. *Bioinformatics*, in press, 2003.