Exploring the Application of Bayesian Networks in Biology

Hongzhu Cui [1], Xinjie Hao[1], Shaocheng Wang[1]
1. Department of Computer Science
Worcester Polytechnic Institute
Worcester, MA, United States
{hcui2, swang3,xhao}@wpi.edu

***Abstract*—In molecular biology, analysis of the gene expression (microarray) data has been proved to be useful to study the regulation of protein synthesis and its reaction to internal and external signals. A Bayesian network, also known as belief networks, is a directed acyclic graph that represents the probabilistic relationship among a set of random variables. The main functionality of Bayesian networks is the capability to express and explain dependencies and uncertainties of a complex problem. Bayesian networks are a promising tool for analyzing gene expression pattern. In this paper, we revisited a previously proposed framework[1] for discovering gene/protein interaction from multiple expression measurements, which is based on the use of Bayesian networks for representing statistical dependencies. We start by showing the typical workflow of Bayesian Network application in reality. We then describe an algorithm (K2)[2] for recovering the network structure from microarray data and two inference algorithms (rejection sampling; likelihood weighting) we applied in this project. Finally, we demonstrate the application of Bayesian Network on the dataset of Sachs *et. al* [3].**

***Keywords* - gene expression; microarray; Bayesian Network; Bioinformatics**

INTRODUCTION AND BACKGROUND

Protein synthesis is regulated by control mechanism at different stages: mRNA transcription, RNA splicing, mRNA transport, translation initialization, etc. One major regulation occurs at the mRNA transcription stage. Recent technology advances in spotting hybridization probes and genome sequencing results in the development of DNA microarrays. DNA microarrays allow biology researchers to measure the abundance of thousands of targeted mRNA at the same time, compared with the traditional method in molecular biology, which could only report the expression level of a single gene. It has been proved that the analysis of microarray is very helpful to understand the gene expression and regulation.[4]

Most of the analysis tools dealing gene expression data are based on clustering algorithm. Although clustering algorithm may provide a compact summarization of data and point to functional relationships between clustered genes, it suffers from some drawbacks. It's based on a global correlation measure. This obscures relationships which exist over only a subset of data. And it cannot detect the interaction between genes different from linear correlation.

In our project, we apply Bayesian network to analyze the gene expression data. A Bayesian network is a graphical representation that expresses probabilistic relationships among a set of random variables. Bayesian Networks are particularly good at describing processes composed of the locally interacting components. And it can also be used to learn causal relationships. Moreover, it can readily handle incomplete dataset and facilitate the combination of domain knowledge and data. For these reasons, Bayesian Network is getting more and more widely used in the fields of genomics, computational molecular biology and bioinformatics. Several previous works[5] indicate that Bayesian networks are a promising tool to gene expression data analysis.

The remainder of this paper organized as follows. In Section 2, we briefly discuss the related work in this field. In Section 3, we describe the problem statement and the work flow of our project. In Section 4, we demonstrate how to learn Bayesian network structure from raw data, and discuss in details about the structure learning algorithm (K2) we used in this project. Also, two inference algorithms we implement will be discussed. In Section 5, we describe the results of our application of Bayesian Network in Sachs et. al data, and evaluate the performance of K2 algorithm by comparing the learned structures with the results got using available Bayesian Network packages. Finally, in Section 6 and Section 7, we conclude with a discussion of K2 algorithm and potential future work.

RELATED WORK

In molecular biology, the most commonly used computational method for analyzing gene expression microarrays is clustering as we mentioned above. But, in our project, we decide to use another method Bayesian network to analyze the gene expression data. There are some researches in this field. We would direct reader to these more comprehensive reviews[4, 6] for current work about Bayesian Network application in analysis of gene expression data.

PROBLEM STATEMENT

A typical workflow in Bayesian Network application consists of three main components: structure learning, parameter learning and inference. The first two problems together are called Bayesian Network learning problem.

The problem of learning a Bayesian Network could be stated as follows. Given a database (D) with multiple cases, we intend to find a network (B) which best fit database D.

The learning of B is usually performed as two-step processes. The first step is structure learning. Its work is to select the most probable Bayesian network model from the cases in D. The second step is parameter learning, which estimate the local distributions implied by the structure learned in the previous step. Thus the Bayesian Network problem then can be expressed by equation 1. Bs denotes the structure of B, and Bp denotes parameter of Bs.

$$P(B|D) = P(Bs, Bp|D) = \underbrace{P(Bs|D)}_{Structure\ Learning} \cdot \underbrace{P(Bp|Bs, D)}_{Parameter\ Learning} \quad (1)$$

Inference on Bayesian networks usually consists of conditional probability (CPQ) queries. Conditional probability queries are concerned with the distribution of a subset of variables Q given some evidence E: CPQ= Prob(Q | E;M).

METHODS

A. Learning the structure of Bayesian Network

Three kinds of algorithms are widely employed to get Bayesian network structures: constraint-based algorithms, score-based algorithms and hybrid algorithms. Constraint-based algorithms[7] utilize statistical tests to learn conditional independence relationships from the data. In score-based algorithms[8], each candidate network is assigned a goodness-of-fit score, which we want to maximise. Hybrid algorithms[9] use conditional independence tests are to restrict the search space for a subsequent score-based search. Considering the limited timeframe, we decide to implement a classic score-based algorithm – K2 algorithm, to learn the Bs of given database.

B. K2 Algorithm

For the purpose of demonstrating K2 algorithm, we come up with a simple example to explain the derivation of score criterion.

Given a database D, which is shown in Table 1.

Based on D, two possible Bayesian network structures Bs1 and Bs2 are shown in Fig.1.

Table 1. A database example D. The term case in the first column denotes a single training instance (record) in the database—as for example, a patient case.

Case	Variable x1	Variable x2	Variable x3
1	present	absent	absent
2	present	present	present
3	absent	absent	present
4	present	present	present
5	absent	absent	absent
6	absent	present	present
7	present	present	present
8	absent	absent	absent
9	present	present	present
10	absent	absent	absent

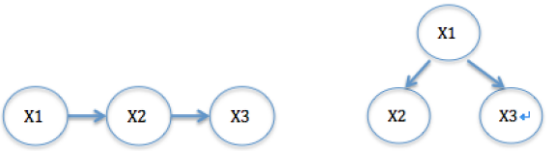


Fig. 1Two potential Bayesian network structures Bs1 and Bs2

By computing such ratios for pairs of belief-network structures, we can rank order a set of structures by their posterior probabilities. As a result, we can compute P(Bs1 | D)/ P(Bs2 | D), in which P(Bsi |D) is the probability of Bsi under the database D. Using Bayesian equation, we can just compute P(Bs1, D)/ P(Bs2, D). Such that P(Bsi | D) can be chosen as score criterion. P(Bsi | D) can be represented by equation 2,

$$P(Bs, D) = \int f(D|Bs, Bp) f(Bp|Bs) P(Bs) dBp \quad (2)$$

where Bp is a vector whose values denote the conditional-probability assignments associated with

Bayesian network structure Bs, and f is the conditional-probability density function over Bp given Bs.

In order to compute P(Bs, D) more efficiently, K2 sets 4 assumptions.

Assumption 1: variables in database should be discrete.

With the statement of this assumption, the equation 2 can be transformed to equation 3.

$$P(Bs, D) = \int P(D|Bs, Bp) f(Bp|Bs) P(Bs) dBp \quad (3)$$

The discrete variables leads us to use the probability mass function P(D|Bs, Bp) in equation 3, rather than the density function f (D | Bs, Bp). As a result, it makes computing with computer possible.

The database used in this project is continuous. To apply the multinomial model, we discretized the data into three levels (Low, Medium or High). Before that, data points that fell more than three standard deviations from the mean were eliminated. However, we can use Hartemink's Discretization algorithm[10] to jointly discretize all the variables. This kind of discretization algorithms will get rid of both normality and linearity assumptions while trying to preserve the dependence structure of the data, which is far better than just using quantile or fixed-length intervals.

Assumption 2: cases in datasets occur independently.

It follows from the conditional independence of cases expressed in assumption 2 that the equation 3 can be transformed to equation 4.

$$P(Bs, D) = \int [\prod_{h=1}^{m} P(Ch|Bs, Bp)] f(Bp|Bs) P(Bs) dBp \quad (4)$$

where m is the number of cases in D and Ch is the hth case in D.

With this assumption, we can separate each case and directly count the number of specific values in a variable from the database case by case.

Assumption 3: no cases have variables missing values

Real-world databases always have missing values, which will affect the execution of K2. Because K2 is score based algorithm, if there are some missing values, the score related to those values are hard to give. For the dataset we applied in this project, we do not encounter missing value problem.

Assumption 4: density function f (Bp | Bs) is uniform

This assumption states that we will indifferently regard the numerical probabilities to place on Bs. In short words, we can get rid of f (Bp | Bs) from equation.

Given assumption 1 through 4, an efficient formula can be presented as equation 5 and 6.

$$N_{ij} = \sum_{k=1}^{r_i} N_{ijk} \quad (5)$$

$$P(Bs, D) = P(Bs) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \quad (6)$$

Suppose each variable x_i in Bs has a set of parents, which can be represented with a list of variables pi_i. Let w_{ij} denote the jth unique instance of pi_i related to D. Suppose there are q_i such unique instance of pi_i. Then N_{ijk} in equation 5 represent the number of cases in D in which variable x_i has the value v_{ik} and pi_i is instantiated as w_{ij}.

P(Bs, D) in equation 6 then can be used as a score criterion. Take Bayesian network structure Bs1 and Bs2 in Fig.1 and database D in Table 1 as an example to show how it works. Applying equation 6 to compute P(Bs1, D) = P(Bs1) * 2.23×10^{-9} like shown in equation 7. Similarly, we can compute P(Bs2, D) = P(Bs2) * 2.23×10^{-10}. If we assume that P(Bs1) = P(Bs2), then P(Bs1, D)/ P(Bs2, D) = 10, which implies Bs1 is 10 times more likely than Bs2 under D.

$$\begin{aligned} P(Bs_1, D) &= P(Bs_1) \frac{(2-1)!\,5!\,5!}{(10+2-1)!} \frac{(2-1)!\,1!\,4!}{(5+2-1)!} \frac{(2-1)!\,4!\,1!}{(5+2-1)!} \frac{(2-1)!\,0!\,5!}{(5+2-1)!} \frac{(2-1)!\,4!\,1!}{(5+2-1)!} \\ &= P(Bs_1) \cdot 2.23 \times 10^{-9} \end{aligned} \quad (7)$$

Finally, K2 is a hill-climbing algorithm to search each variable's "best" parents list π using the score criterion shown in equation 8 based on equation 6.

$$g(i, \pi_i) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \quad (8)$$

Fig. 2 shows the Pseudo-code of K2, one more thing need to noted is that the ordering on the nodes is a permutation of all nodes n1, n2,…nk, the parent of node ni can only be selected from nj that proceed ni. The original paper does not suggest a way of arranging the ordering. In our project, we followed the validated network to arrange the ordering manually, which could be improved some recent proposed method[11].

```
Pseudo-code of K2:

{Input: A set of n nodes, an ordering on the nodes,
 an upper bound u on the number of  parents a node
 may have, and a database D containing m cases}

{Output: The parents node list of each node}

For each node i:
    set its parents list πi = ø
    record P_max = g(i, πi)

    While(Didn't check all the nodes j preceed i in
     the ordering && didn't beyond the upper bound):

        If g(i, {πi} ∪ j) > P_max:
            P_max = g(i, {πi} ∪ j)
            add node j to πi
        end If

    end While

end For

Return each πi
```

Fig. 2: Pseudo-code of K2

C. Parameter learning and Inference

Due to time limit, for parameter learning work, we resort to the R Bayeasian Network learning package – bnlearn[12].

We implemented two inference algorithms we've learned in this project. First, we write the code to read the file containing Bayesian network information. Then it constructs the Bayesian network. We implemented rejection sampling algorithm, generated many samples by using the CPT and their relationships. Then check if these samples are satisfying the given variables. We only took those samples which satisfy the given variable. And calculate the probability statistically, it turns out the result is close to the result from Macro Scutari. We also implemented likelihood weighting algorithm, this algorithm only generates legal samples which satisfy the given variables. It assigns each sample a weight by likelihood it satisfies the given state. We then evaluate its results. The results from these two algorithms converged under same given variables. These two algorithms can both return consistent estimates, but rejection sampling is expensive if the probability of given variable is too small. For the likelihood weighting algorithm, since this Bayesian network doesn't have too many variables, so no total weight exist in these samples.

RESULTS

We applied our approach to the dataset of Sachs et. al. The corresponding paper highlights the use of interventional data, compared with observational data. And the results are validated using existing literature. The data consists of the simultaneous measurements of 11 phosphorylated proteins and phospholypids derived from thousands of individual primary immune system cells. 1200 data points subject only to general stimulatory cues, so that the protein signaling paths are active; and such data is called observational. In addition to the 1200 data set, we have 9 other data sets subject to different targeted stimulatory cues and inhibitory interventions. Such data are often called interventional. Overall, the interventional data set contains 5400 observations with no missing values.

D. Structure learning results:

To evaluate the performance of the K2 algorithm we applied in our project, we compared our learned network structure (Fig. 3) to the validated network using existing literature (Fig. 4) and the network structures learned using two popular Bayesian Network learning tool/package: bnlearn and BNFinder[13]. (Fig. S1 and Fig. S2). Besides Fig. S2 was visualized in Cytoscape[14],all the others are visualized in R using R package Rgraphviz[15].

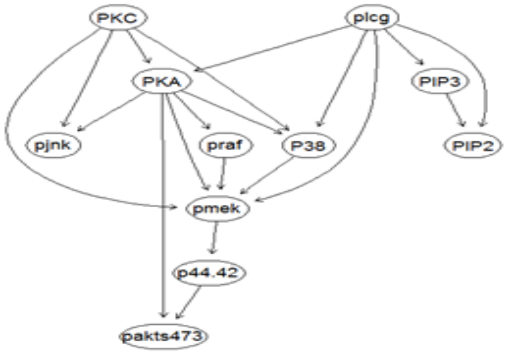


Fig. 3 Network structure learned using K2 algorithm

Fig. 4 validated network using existing literature

As we could see from Fig. 2, the validated network contains 11 nodes and 17 edges. For our structure learning results inferred from 5400 interventional data points as shown in Fig. 1, the topology is quite consistent with the literature validated. Out of 17 expected edges,

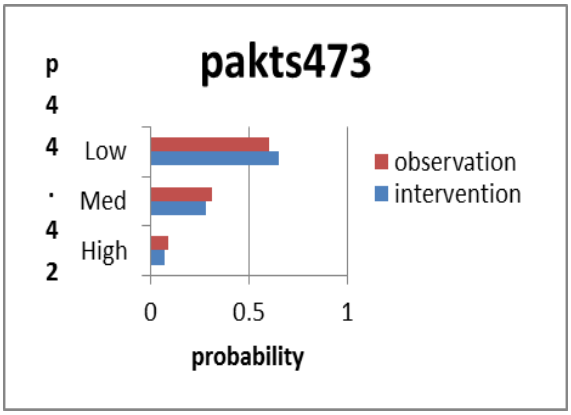
our implementation of K2 recovered 11 correctly. Compared with the results we got using bnlearn(13 recovered, 7 unexplained) and BNFinder(11 recovered), the results is competitive against current available Bayesian Network tool and confirms K2's ability of recovering the underlying network structure.

The main problem with K2 algorithm is that K2 reported more unexplained edges than bnlearn and BNFinder. K2 introduced 7 edges unreported in literature. Especially, it connected plcg to three other proteins in the network.

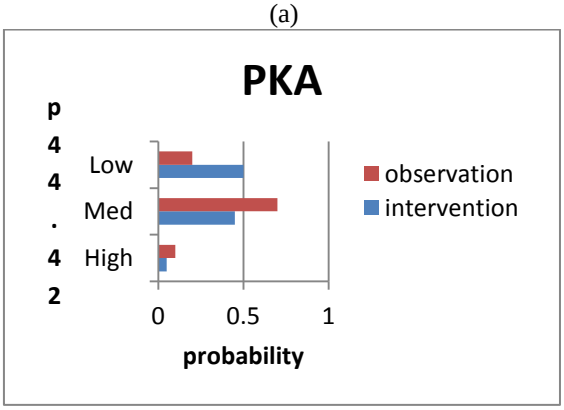
The K2 algorithm was also performed on the observational data. The network learned from 1200 observational data points contains only 10 edges. 7 of them are expected, and 10 edges were missing. Such an observation verifies the claim in the paper that intervention plays an important role in effective inference, especially in establishing the directionality of the connections. Moreover, we applied the K2 algorithm on a truncated single-cell (840 data points), whose size is comparable to the simulated Western blot data set. We excluded most of the data in the original data set randomly to reduce the size. However, the inference result on the truncated data set shows an obvious decline of inference accuracy. Compared to the 5400 data points result, we missed more connections and reported more unexpected edges. The original paper also noted that sufficiently large data set size is critical in network inference, which is consistent with our findings.

E. Inference Results

In their paper, Sachs et al. made two claims based on the validated network: 1). a direct perturbation of p44.42 should influence pakts473; 2). a direct perturbation of p44.42 should not influence PKA. They validated these two claims using both the probability distributions of p44.42, pakts473 and PKA and the results of two ad-hoc experiments. Here, we applied two approximate inference algorithms: rejection sampling and likelihood weighting to calculate the conditional probability distribution of pakts473 (Prob(pakts473|p44.42)) and PKA (Prob(PKA|p44.42)) in both observational and interventional cases. The inference results from the two algorithms are very close. The below figure (Fig. 5) shows the probability distribution we got from rejection sampling algorithm. From the above figure, we could tell that under different conditions (observation or intervention), the conditional distribution of P(PKA|p44.42) varies a lot, compared with P(pakts473|p44.42), which is more stable. And this result validates the claim in the paper.



(a)



(b)

Fig. 5 Conditional probability table calculated for pakts473 and PKA using rejection sampling algorithm

DISCUSSION

K2 algorithm we apply in this project has been shown effective in structure learning. But it could be optimized in several ways. First, during the execution of K2, the factorial calculation in g(i, π) is time consuming. However, if we modify computing g into computing log(g), without losing correctness, we transform multiply and division operations to addition and subtraction operations, which will save a lot of running time. Second, the ordering of nodes in input of K2 is to prevent creating loops in Bs we get. But the Bs we get as a result relies much on the correctness of the ordering. So a good algorithm to get satisfied ordering is our future work. Third, as K2 is a hill-climbing algorithm. It is easy to get stuck in local maxima because whenever you add a node into parent list, it cannot be removed anymore. In our project, the misleading of the node 'plcg' has a high relationship with this problem. Because node 'plcg' in the relative beginning of the ordering, it is easy to be added into parent list of other nodes when the real powerful parent nodes have not been considered yet. So

we could use heuristic or randomized method to help K2 get the chance to get rid of these noising nodes.

CONCLUSION

In summary, the Beyesian network framework we implement could successfully recover the underlying network structure and regulation relationship between different genes. It confirms that the Bayesian network is a useful approach for analyzing gene expression data. Structure learning is a very important part in Bayesian network work flow. And our work verified that K2 algorithm is an effective algorithm for structure learning. However, it still could be optimized in several ways. The inference algorithms we used are classic methods the inference purpose. They are easy to implement and can return consistent estimates. But rejection sampling is expensive if the probability of given variable is too small. Since the Bayesian network for this problem doesn't have too many nodes, so the likelihood weighting may be a better inference algorithm in this case.

REFERENCE

[1] C. J. Needham, J. R. Bradford, A. J. Bulpitt, and D. R. Westhead, "A primer on learning in Bayesian networks for computational biology," *PLoS computational biology*, vol. 3, p. e129, 2007.

[2] G. F. Cooper and E. Herskovits, "A Bayesian method for the induction of probabilistic networks from data," *Machine learning*, vol. 9, pp. 309-347, 1992.

[3] K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan, "Causal protein-signaling networks derived from multiparameter single-cell data," *Science*, vol. 308, pp. 523-529, 2005.

[4] D. J. Wilkinson, "Bayesian methods in bioinformatics and computational systems biology," *Briefings in bioinformatics*, vol. 8, pp. 109-116, 2007.

[5] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using Bayesian networks to analyze expression data," *Journal of computational biology*, vol. 7, pp. 601-620, 2000.

[6] M. Hecker, S. Lambeck, S. Toepfer, E. Van Someren, and R. Guthke, "Gene regulatory network inference: data integration in dynamic models—a review," *Biosystems*, vol. 96, pp. 86-103, 2009.

[7] D. Edwards, *Introduction to graphical modelling*: Springer, 2000.

[8] P. Larranaga, B. Sierra, M. J. Gallego, M. J. Michelena, and J. M. Picaza, "Learning Bayesian networks by genetic algorithms: a case study in the prediction of survival in malignant skin melanoma," in *Artificial Intelligence in Medicine*, ed: Springer, 1997, pp. 261-272.

[9] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," *Machine learning*, vol. 65, pp. 31-78, 2006.

[10] A. J. Hartemink, "Principled computational methods for the validation and discovery of genetic regulatory networks," Citeseer, 2001.

[11] Z. Wei, H. Xu, W. Li, X. Gui, and X. Wu, "Improved Bayesian Network Structure Learning with Node Ordering via K2 Algorithm," in *Intelligent Computing Methodologies*, ed: Springer, 2014, pp. 44-55.

[12] M. Scutari, "'bnlearn'-an R package for Bayesian network learning and inference," *UCL Genetics Institute, University College, London, London, UK*, 2011.

[13] B. Wilczyński and N. Dojer, "BNFinder: exact and efficient method for learning Bayesian networks," *Bioinformatics*, vol. 25, pp. 286-287, 2009.

[14] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, *et al.*, "Cytoscape: a software environment for integrated models of biomolecular interaction networks," *Genome research*, vol. 13, pp. 2498-2504, 2003.

[15] F. Hahne, "Rgraphviz: A new interface to render graphs using Rgraphviz," ed, 2012.

Supplementary materials:

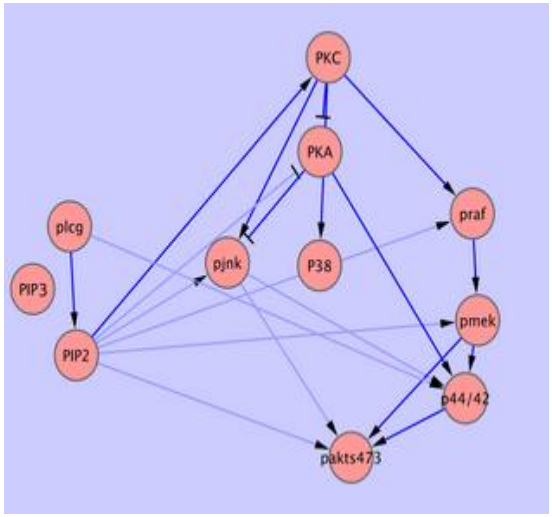


Fig. S1. Network recovered using BNFinder

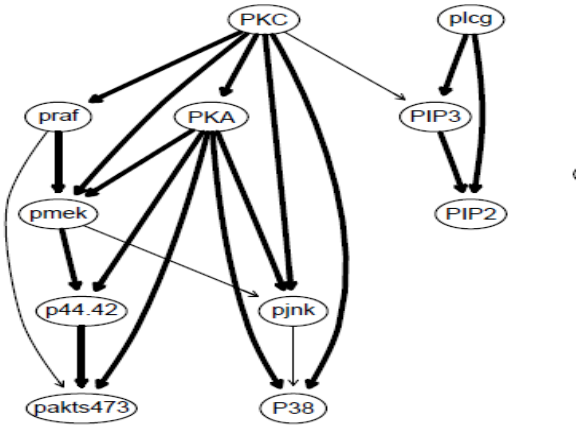


Fig. S2. Network recovered using bnlearn