BIOINFORMATICS APPLICATIONS NOTE Vol. 27 no. 19 2011, pages 2765–2766
doi:10.1093/bioinformatics/btr457

Systems biology

Advance Access publication August 3, 2011

GlobalMIT: learning globally optimal dynamic bayesian network with the mutual information test criterion

Nguyen Xuan Vinh[1,*], Madhu Chetty[1,*], Ross Coppel[2] and Pramod P. Wangikar[3]

[1]Gippsland School of Information Technology, Faculty of IT, Monash University, [2]Department of Microbiology, Faculty of Medicine, Nursing and Health Sciences, Monash University, Victoria, Australia and [3]Department of Chemical Engineering, Indian Institute of Technology, Bombay, India

Associate Editor: Trey Ideker

ABSTRACT

Motivation: Dynamic Bayesian networks (DBN) are widely applied in modeling various biological networks including the gene regulatory network (GRN). Due to the NP-hard nature of learning static Bayesian network structure, most methods for learning DBN also employ either local search such as hill climbing, or a meta stochastic global optimization framework such as genetic algorithm or simulated annealing.

Results: This article presents GlobalMIT, a toolbox for learning the globally optimal DBN structure from gene expression data. We propose using a recently introduced information theoretic-based scoring metric named mutual information test (MIT). With MIT, the task of learning the globally optimal DBN is efficiently achieved in polynomial time.

Availability: The toolbox, implemented in Matlab and C++, is available at http://code.google.com/p/globalmit.

Contact: vinh.nguyen@monash.edu; madhu.chetty@monash.edu

Supplementary information: Supplementary data is available at *Bioinformatics* online.

Received on June 8, 2011; revised on July 21, 2011; accepted on July 29, 2011

1 INTRODUCTION

Bayesian network (BN) has found applications in modeling various biological networks including the gene regulatory network (GRN). The two important limitations when applying static BN to these domain problems are: (i) BN does not have a mechanism for exploiting the temporal aspect of time-series data, such as time-series microarray data; and (ii) BN does not allow the modeling of cyclic phenomena, such as feedback loops, which are prevalent in biological systems (Yu *et al.*, 2004). These drawbacks have motivated the development of the so-called dynamic Bayesian network (DBN). Its simplest model, the first-order Markov stationary DBN, assumes that both the structure of the network and the parameters characterizing it remain unchanged over time. The value of a variable at time (t) is assumed to depend only on the value of its parents at time ($t-1$). DBN not only accounts for the temporal aspect of time-series data (i.e. an inter time-slice edge must always be directed forward in time), but it also allows the modeling of feedback loops. Since its inception, DBN has received particular interest from the bioinformatics community (Husmeier, 2003; Kim *et al.*, 2003; Murphy and Mian, 1999; Perrin *et al.*, 2003; Wilczynski and Dojer, 2009; Yu *et al.*, 2004; Zou and Conzen, 2005).

*To whom correspondence should be addressed.

2 METHOD

Most algorithms to date for learning DBN structure employ a local search strategy, such as hill climbing with random restart, or a meta stochastic global optimization framework such as genetic algorithm or simulated annealing, as exemplified by several softwares such as BANJO (Smith *et al.*, 2006) or bnlearn (Scutari, 2010). This is due to several NP-hardness results in learning static BN structure (see, e.g. Chickering, 1996). Recently, Dojer (2006) has shown otherwise that learning DBN structure, as opposed to static BN, does not necessarily have to be NP-hard. In particular, it was shown that, under some mild assumptions, there are algorithms using the minimum description length (MDL) and BDe scores, which find the globally optimal network with a polynomial worst-case time complexity. These algorithms have been realized within the BNFinder software (Wilczynski and Dojer, 2009). In our experiments, we observed that BNFinder+MDL is very fast, whereas BNFinder+BDe is very time demanding: a single run on a dataset of 20 genes and 300 observations can take up to a day (Vinh *et al.*, 2011). This is in concordance with the theoretical worst-case complexity analysis, where the algorithm would have to exhaustively evaluate all possible parent sets of cardinality from 0 to p^*-1. Let k be the number of discrete states of each variable, N be the number of experiments, then for MDL, p^*_{MDL} is given by $\lceil \log_k N \rceil$, while for BDe, $p^*_{\text{BDe}} = \lceil N \log_{\gamma^{-1}} k \rceil$, where $0 < \gamma < 1$ is the network complexity penalty parameter (default value $\log \gamma^{-1} = 1$ for BNFinder). In general, p^*_{BDe} scales linearly with the number of data items N, making its value of less practical interest, even for very small datasets. Although being more expensive, BNFinder+BDe is still recommended over BNFinder+MDL, '*due to its exactness in the statistical interpretation*' (Wilczynski and Dojer, 2009). Further, de Campos (2006) also showed that BDe seems to learn more accurate networks than MDL [which is also equivalent to the Bayesian Information Criterion (BIC)].

Mutual information test (MIT) is a recently introduced scoring metric for learning BN (de Campos, 2006). To understand MIT, let $\mathbf{X} = \{X_1, \ldots, X_n\}$ denote the set of n variables with corresponding $\{r_1, \ldots, r_n\}$ discrete states, D denote our dataset of N observations, G be a DBN, and $\mathbf{Pa}_i = \{X_{i1}, \ldots, X_{is_i}\}$ be the set of parents of X_i in G with corresponding $\{r_{i1}, \ldots, r_{is_i}\}$ discrete states, $s_i = |\mathbf{Pa}_i|$. The MIT score is then defined as:

$$S_{\text{MIT}}(G:D) = \sum_{i=1;\mathbf{Pa}_i \neq \emptyset}^{n} \{2N \cdot I(X_i, \mathbf{Pa}_i) - \sum_{j=1}^{s_i} \chi_{\alpha, l_i \sigma_i(j)}\}$$

where $I(X_i, \mathbf{Pa}_i)$ is the mutual information between X_i and its parents as estimated from D. $\chi_{\alpha, l_{ij}}$ is the value such that $p(\chi^2(l_{ij}) \leq \chi_{\alpha, l_{ij}}) = \alpha$ (the chi-square distribution at significance level $1-\alpha$), and the term $l_{i\sigma_i(j)}$ is defined as:

$$l_{i\sigma_i(j)} = \begin{cases} (r_i - 1)(r_{i\sigma_i(j)} - 1) \prod_{k=1}^{j-1} r_{i\sigma_i(k)}, & j = 2 \ldots, s_i \\ (r_i - 1)(r_{i\sigma_i(j)} - 1), & j = 1 \end{cases}$$

where $\sigma_i = \{\sigma_i(1), \ldots, \sigma_i(s_i)\}$ is any permutation of the index set $\{1 \ldots s_i\}$ of $\mathbf{Pa}_i$, with the first variable having the greatest number of states, the second variable having the second largest number of states, and so on. MIT falls under the same category of information theory-based scores as the MDL, BIC and Akaike Information Criterion (AIC). Briefly speaking, under MIT, the goodness-of-fit of a network is measured by the total mutual information shared between each node and its parents, penalized by a term which quantifies the degree of statistical significance of this shared information. Through extensive experimental validation, de Campos (2006) suggested that, for the task of learning static BN, MIT can compete favorably with Bayesian scores (BDe), outperforms BIC/MDL and should be the score of reference within those based on information theory. However, as opposed to the other popular scoring metrics, to our knowledge MIT has not been considered for DBN learning.

In our recent work (Vinh *et al.*, 2011), we have shown that under the same set of assumptions made in Dojer (2006), there exists a polynomial worst-case time complexity algorithm for learning the globally optimal DBN structure with MIT. We call this algorithm GlobalMIT. The polynomial worst-case time complexity of GlobalMIT is characterized by:

$$p^*_{\mathrm{MIT}} = \arg\min\{p | \sum_{j=1}^{p} \chi_{\alpha, l_{i\sigma_i(j)}} \geq 2N \cdot \log k\},$$

It can be seen that p^*_{MIT} depends only on α, k and N. In the worst case, our algorithm will have to examine all the possible parent sets of cardinality from 1 to $p^*_{\mathrm{MIT}} - 1$. Since there are $O(n^{p^*_{\mathrm{MIT}}})$ subsets with at most $p^*_{\mathrm{MIT}} - 1$ parents, and each set of parents can be scored in polynomial time, globalMIT admits an overall polynomial worst-case time complexity in the number of variables (see also *GlobalMIT user guide* within the online supplementary material for further details). Our experimental evaluation in Vinh *et al.* (2011) showed that GlobalMIT is very competitive in terms of network quality. In other words, GlobalMIT seems to combine the strength of both MDL (speed) and BDe (solution quality).

2.1 Implementation

The algorithm involves investigating, for each variable, every potential parent set of increasing cardinality (not exceeding $p^*_{\mathrm{MIT}} - 1$) until the globally optimal solution has been found. One important observation for the efficient implementation of GlobalMIT is the following decomposition property of the mutual information:

$$I(X_i, \mathbf{Pa}_i \cup X_j) = I(X_i, \mathbf{Pa}_i) + I(X_i, X_j | \mathbf{Pa}_i).$$

This implies that the mutual information can be computed incrementally, and suggests that, for efficiency, the computed mutual information values should be cached to avoid redundant computations.

We provide an implementation of the GlobalMIT algorithm as a Matlab toolbox. The toolbox also supports simple data pre-processing functionalities, such as data discretization and mapping, and simple post-processing such as visualization and quality assessment. For improved performance, we also provide an implementation of GlobalMIT in C++. Our experiments showed that the C++ version of GlobalMIT is up to 40 times on average faster than the Matlab version. For seamless and easy use of GlobalMIT, interface modules provide connection between Matlab and the C++ version, allowing the users to perform all operations in Matlab.

We experimentally compared the runtime of GlobalMIT to BNFinder (Wilczynski and Dojer, 2009) with both the MDL and BDe metrics. The test was carried out on a synthetic dataset of 20 genes $\times$ 2000 observations, generated from a gene regulatory network (Network No. 1) as described in Yu *et al.* (2004). On a Core 2 Duo PC with 4 GB of main memory, GlobalMIT C++ and BNFinder + MDL took slightly >1 h to analyze this dataset, while BNFinder+BDe took >3 days.

It is noted that currently GlobalMIT only learns DBN with inter-time slice edges, i.e. edges from $X_i^{(t-1)}$ to $X_j^{(t)}$. Learning DBN which allows both inter-and intratime slide edges falls back to an NP-hard problem. However, intratime slice edges representing (almost) instantaneous genetic interactions, if of interest, can be learned separately using some BN learning algorithm, then combined with the intertime slice edges, followed by some post-processing for the final result. Our future work includes expanding GlobalMIT in this direction. Also, we are extending our framework to handle edges spanning either two or several time slices, which represent variable, longer time-delayed genetic interactions that are also abundant in genetic networks (Zou and Conzen, 2005).

Funding: This research forms part of a project supported by an Australia-India strategic research fund (AISRF).

Conflict of Interest: none declared.

REFERENCES

Chickering,D.M. (1996) Learning Bayesian networks is NP-complete. In Fisher,D. and Lenz,H. (eds) *Learning from Data: Artificial Intelligence and Statistics V*, pp. 121–130.

de Campos,L.M. (2006) A scoring function for learning bayesian networks based on mutual information and conditional independence tests. *J. Mach. Learn. Res.*, **7**, 2149–2187.

Dojer,N. (2006) Learning Bayesian networks does not have to be NP-hard. In Královic,R. and Urzyczyn,P. (eds) *Mathematical Foundations of Computer Science*, Vol. 4162 of *Lecture Notes in Computer Science*. Springer, Berlin/Heidelberg, pp. 305–314.

Husmeier,D. (2003) Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks. *Bioinformatics*, **19**, 2271–2282.

Kim,S.Y. *et al.* (2003) Inferring gene networks from time series microarray data using dynamic Bayesian networks. *Brief. Bioinformat.*, **4**, 228–235.

Murphy,K.P. and Mian,S. (1999) Modelling gene expression data using dynamic bayesian networks. *Technical Report*, Computer Science Division, University of California, Berkeley, CA.

Perrin,B.-E. *et al.* (2003) Gene networks inference using dynamic Bayesian networks. *Bioinformatics*, **19** (Suppl. 2), ii138–ii148.

Scutari,M. (2010) Learning Bayesian networks with the bnlearn R package. *J. Stat. Soft.*, **35**, 1–22.

Smith,V.A. *et al.* (2006) Computational inference of neural information flow networks. *PLoS Comput. Biol.*, **2**, e161.

Vinh,N.X. *et al.* (2011) A polynomial time algorithm for learning globally optimal dynamic Bayesian network and its applications in genetic network reconstruction. *Technical Report FIT-GSIT TR.1101*, Faculty of IT, Monash University.

Wilczynski,B. and Dojer,N. (2009) BNFinder: exact and efficient method for learning Bayesian networks. *Bioinformatics*, **25**, 286–287.

Yu,J. *et al.* (2004) Advances to Bayesian network inference for generating causal networks from observational biological data. *Bioinformatics*, **20**, 3594–3603.

Zou,M. and Conzen,S.D. (2005) A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. *Bioinformatics*, **21**, 71–79.