Bayesian Network Learning and Applications in Bioinformatics

By

Xiaotong Lin

Submitted to the Department of Electrical Engineering and Computer Science and the
Faculty of the Graduate School of the University of Kansas
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Committee members

Dr. Jun Huan, Chairperson

Dr. Swapan Chakrabarti

Dr. Bo Luo

Dr. Brian Potetz

Dr. Robert E. Ward

Date defended: July 26, 2012

The Dissertation Committee for Xiaotong Lin certifies
that this is the approved version of the following dissertation :

Bayesian Network Learning and Applications in Bioinformatics

Dr. Jun Huan, Chairperson

Date approved: July 26, 2012

Acknowledgements

I am grateful for my adviser Dr. Jun Huan, who has been supportive academically and financially over the years. I am also indebted to Dr. Bo Luo , Dr. Swapan Chakrabarti, Dr. Brian Potetz, and Dr. Robert Ward for their serving on my committee. Finally, the thesis is impossible without the love and consistent support of my family: Jeffrey, Jessica and Xue-wen.

Abstract

A Bayesian network (BN) is a compact graphic representation of the probabilistic relationships among a set of random variables. The advantages of the BN formalism include its rigorous mathematical basis, the characteristics of locality both in knowledge representation and during inference, and the innate way to deal with uncertainty. Over the past decades, BNs have gained increasing interests in many areas, including bioinformatics which studies the mathematical and computing approaches to understand biological processes.

In this thesis, I develop new methods for BN structure learning with applications to biological network reconstruction and assessment. The first application is to reconstruct the genetic regulatory network (GRN), where each gene is modeled as a node and an edge indicates a regulatory relationship between two genes. In this task, we are given time-series microarray gene expression measurements for tens of thousands of genes, which can be modeled as true gene expressions mixed with noise in data generation, variability of the underlying biological systems etc. We develop a novel BN structure learning algorithm for reconstructing GRNs.

The second application is to develop a BN method for protein-protein interaction (PPI) assessment. PPIs are the foundation of most biological mechanisms, and the knowledge on PPI provides one of the most valuable resources from which annotations of genes and proteins can be discovered. Experimentally, recently-developed high-throughput technologies have been carried out to reveal protein interactions in many organisms. However, high-throughput interaction data often contain a large number of

spurious interactions. In this thesis, I develop a novel *in silico* model for PPI assessment. Our model is based on a BN that integrates heterogeneous data sources from different organisms.

The main contributions are:

1. A new concept to depict the dynamic dependence relationships among random variables, which widely exist in biological processes, such as the relationships among genes and genes' products in regulatory networks and signaling pathways. This concept leads to a novel algorithm for dynamic Bayesian network learning. We apply it to time-series microarray gene expression data, and discover some missing links in a well-known regulatory pathway. Those new causal relationships between genes have been found supportive evidences in literature.

2. Discovery and theoretical proof of an asymptotic property of K2 algorithm (a well-known efficient BN structure learning approach). This property has been used to identify Markov blankets (MB) in a Bayesian network, and further recover the BN structure. This hybrid algorithm is evaluated on a benchmark regulatory pathway, and obtains better results than some state-of-art Bayesian learning approaches.

3. A Bayesian network based integrative method which incorporates heterogeneous data sources from different organisms to predict protein-protein interactions (PPI) in a target organism. The framework is employed in human PPI prediction and in assessment of high-throughput PPI data. Furthermore, our experiments reveal some interesting biological results.

4. We introduce the learning of a TAN (Tree Augmented Naïve Bayes) based network, which has the computational simplicity and robustness to high-throughput PPI assessment. The empirical results show that our method outperforms naïve Bayes and a manual constructed Bayesian Network, additionally demonstrate sufficient information from model organisms can achieve high accuracy in PPI prediction.

Contents

List of Figures

List of Tables

1. Introduction

1.1 Motivation

A Bayesian network (BN) is a probabilistic graphical model that characterizes a joint probability distribution among a set of random variables, using a directed acyclic graph (DAG) and a conditional probability table (CPT) for each variable in the network given its parent set. The graphical representation visually captures conditional dependencies among the variables and allows for encoding expert knowledge in uncertain domains. Bayesian networks also provide a natural way to incorporate heterogeneous data into a single model and integrate existing knowledge with new informations. Therefore, Bayesian networks have been widely applied to a variety of fields.

In the medical field, a Bayesian network has long been used for diagnosis, prognosis, and treatment selection [1–7]. In artificial intelligence area, Bayesian networks have been used in natural spoken dialog systems [8], vision recognition [9], expert system [10]. Recently, Bayesian networks have been used in data mining, search engine optimization, computational molecular Biology, Bioinformatics [11], and biological data integration.

There are three main tasks for BNs: probabilistic inference, parameter learning, and structure learning [12]. Inference refers to the task of computing the posterior distribution over a set of query variables given some evidence variables in the network. It is only possible if we know in advance the network structures and the associated CPT. Parameter learning also assumes that the network structure is available and focuses on predicting the probability distribution for every variable conditional on its parents in the network. Apparently, the most fundamental task in applying BNs is to construct the network structures.

In some simple applications, the network structure of a DAG is manually constructed by experts and is then used for inference. For example, one can build a belief network by thinking in terms

of causal relationships between diseases and symptoms (e.g., fatigue, loss of appetite, sore throat, cold, strep throat, or something more serious). There are many other applications, however, which are either time consuming or too complex for experts to build a network. Alternatively, one may construct the network structure from data via computational learning methods.

Bayesian networks are a succinct and efficient way to represent a joint probability distribution among a set of variables. Besides their ability for density estimation [13], their semantics lend them to what is sometimes loosely referred to as causal discovery, namely directional relationships among quantities involved. It has been widely accepted that the most parsimonious representation for a Bayesian net is one that closely represents the causal independence relationships that may exist. For these reasons, there has been great interest in automatically inducing the structure of Bayesian nets from data, preferably also preserving the independence relationships in the process.

1.2 Problem Statement

In this thesis, I will focus on developing effective and efficient methods for learning structures of Bayesian networks and applying the new methods for biological data analyses. My motivations for structure learning are (1) to learn a model for reconstructing domain structures (e.g., genetic regulatory networks) and (2) to use the model for classification tasks (e.g., assessment of protein interactions). I will address two significant problems of BN structure learning, risen in bioinformatics applications, namely limited sample size and high dimensionality. For example, in microarray-based genetic regulatory network reconstruction tasks, one typically has, on one hand, less than 200 instances and on the other hand, several thousands of genes. As a result, small sample sizes tend to introduce many false positives and high dimensionality leads to computationally intractable problems.

2. Background and Related Work

2.1 Bayesian Networks

A Bayesian network is a probabilistic graphical model that typically consists of two components: (1) a DAG with nodes representing random variables and edges characterizing causal relationships between nodes, and (2) a probability table that describes the conditional dependencies among nodes. Fig. 2.1 shows an example of a BN with five binary nodes (variables) and five edges.

The *necessary condition* for a DAG $\mathscr{G}$ to be a Bayesian network of a probability distribution $\mathscr{P}$ is for $\mathscr{P}$ to admit the product decomposition dictated by $\mathscr{G}$, as given below [14] :

$$P(X_1, \cdots, X_n) = \prod_i P(X_i|\pi(X_i)) \tag{2.1}$$

Where, $X_1, \cdots, X_n$ are n random variables in the distribution $\mathscr{P}$, also n nodes in DAG $\mathscr{G}$. $\pi(X_i)$ is the set of nodes(variables) which directly precede Node X_i, namely, the parent set of X_i. The left side of Equation 2.1 is the joint probability of the n random variables, and $P(X_i|\pi(X_i))$ is the probability of X_i conditioned on its parent set. Equation 2.1 is also called the decomposition property of a Bayesian network. If a distribution and a DAG satisfy the above relationship, they are said compatible or Markov related. The decomposition property is the theoretical foundation of Bayesian network inference and greatly simplifies the process.

In this work, we only consider discrete variables and complete datasets (i.e., no missing data). In the following definitions, I will use an upper-case letter to represent a variable (e.g., X) and the same lower-case letter to denote a state of that variable (e.g., x). The structure of a BN represents the conditional independence among variables, which is defined as follows:

Definition 1 (conditionally independent) [15]: Considering three distinct variables X, Y and Z in a variable set V, random variables X and Y are conditionally independent given Z if $\forall x, y, z$ and

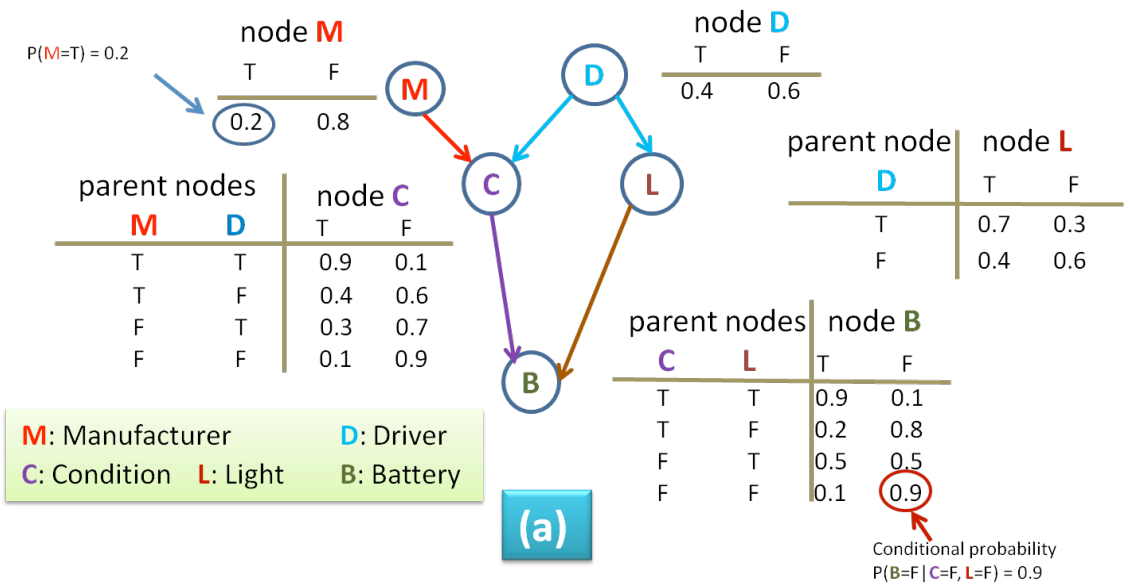


Figure 2.1 – A synthetic Bayesian network with a directed acyclic graph and conditional probability tables, where each value represents the conditional probability of a node given its parent nodes.

$P(Z = z) > 0$,

$$P(X = x, Y = y|Z = z) = P(X = x|Z = z) \times P(Y = y|Z = z) \tag{2.2}$$

This conditional independence is denoted as $(X \perp Y|Z)$.

The relationship between the conditional independence and certain graphical structures can be described by a DAG property called *d-separation*, which is defined on the basis of paths:

Definition 2 (d-separation) [14]: A path p is said to be d-separated (or blocked) by a set of nodes Z if and only if

1. p contains a chain $i \to m \to j$ or a fork $i \leftarrow m \to j$, such that the middle node m is in Z, or
2. p contains an inverted fork (or collider) $i \to m \leftarrow j$, such that the middle node m is not in Z and no descendant of m is in Z

A set Z is said to d-separate X from Y if and only if Z blocks every path from a node in X to a node in Y, denoted as $(X \nleftrightarrow Y|Z)$. If X and Y are not d-separated by Z, they are called *d-connected*, written as $(X \leftrightarrow Y|Z)$.

The concept of d-separation is often used in graphically testing whether a variable X is inde-

pendent of another variable Y given a set of variables Z in a DAG. This practice is based on the assumption of the *faithfulness condition* (defined next) between a DAG and a probability distribution.

Definition 3 (faithfulness condition) [15]: Suppose we have a joint probability distribution $\mathscr{P}$ of the random variables in a set V and a DAG $\mathscr{G} = (V, E)$ (V is the set of vertices, E is a set of directed edges). Then $(\mathscr{P}, \mathscr{G})$ satisfy the faithfulness condition if and only if all and only conditional independencies in $\mathscr{P}$ are identified by d-separation in $\mathscr{G}$. $\mathscr{G}$ is called a perfect map of $\mathscr{P}$ if they satisfy the faithfulness condition.

Definition 4 (Markov Blanket) [15]: Let V be a set of random variables, P be their joint probability distribution, and $X \in V$. Then a Markov blanket $Mb(X)$ is any set of variables such that X is conditionally independent of all the other variables given $Mb(X)$. That is,

$$X \perp \{V - X - Mb(X)\} | Mb(X) \tag{2.3}$$

Markov boundary of X is defined as any Markov blanket such that none of its proper subsets is a Markov blanket of X. In other words, Markov boundary is the smallest Markov blanket. Thereafter in this thesis, when Markov blanket is mentioned, we actually mean Markov boundary.

In a Bayesian network $(\mathscr{G}, \mathscr{P})$ where $\mathscr{G}$ and $\mathscr{P}$ satisfy the faithfulness condition, the Markov blanket of a node X is the set of its parents, children and children's parents(spouse).

As illustrated in Figure 2.2, the Markov Blanket of the node X , $Mb(X)$, contains its parents (B and C), children (D and E) and the direct parent of the direct child (A - parent of D), i.e., $Mb(X) = \{B, C, D, E, A\}$. When given $Mb(X)$, X is independent to the outside nodes $Z = \{U, V, W, S\}$, or $X \perp Z | Mb(X)$.

2.2 Struture Learning Methods

Without knowing the structure and the probability table of a Bayesian network in advance, the goal of structure learning is to reconstruct the network based on the data available. A fundamental as-

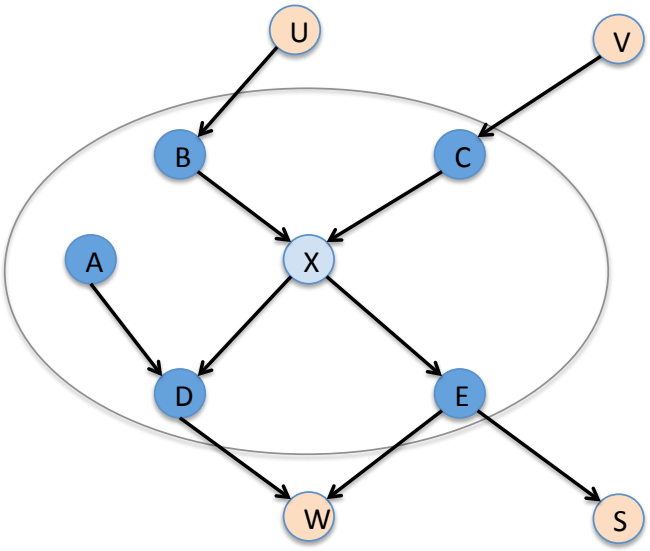


Figure 2.2 – An example of Markov Blanket in a Bayesian network: the Markov Blanket of the node X , $Mb(X)$ contains its parents (B and C), children (D and E) and the direct parent of the direct child (A, parent of D), i.e., $Mb(X) = \{B, C, D, E, A\}$

sumption in structure learning is that the dataset contains independently and identically distributed (*i.i.d.*) instances generated from an underlying distribution $\mathscr{P}$, which is induced by a Bayesian network $\mathscr{G}$. Following this assumption, a number of computational methods have been developed for structure learning, which can be roughly grouped into three classes: constraint-based, search-and-score based, and hybrid methods.

2.2.1 Constraint-based methods

Constraint-based methods try to find a DAG by exploring the independence in a BN [12]. They run some conditional dependence and independence tests in the training data and construct a network that is consistent with these tests (constraints) [16–24]. For example, Geiger et al. [17] developed a recovery algorithm to reconstruct a simplified DAG, called polytree (a singly-connected DAG that satisfies three properties: composition, intersection and marginal weak transitivity), in a polynomial time. They assumed a data distribution that was close to Gaussian. Meek [19] proposed a dependency model-based method (a dependency model is a set of independence statements), which consists of several steps including building an undirected graph first and assigning directions for

edges. The background knowledge was also introduced for extending the DAG. de Campos and Huete [20] proposed an independence test- based approach to learn a special class of DAGs with priori limiting structures , called simple graphs. A simple graph is a DAG satisfying the following condition: two couple nodes (nodes sharing a common direct child) are d-separated by the empty set. Although structurally limited, simple graphs represent richer sets of independence relationships than polytrees (indeed, polytrees can be treated as a special case of simple graphs). The structure recovery algorithm developed by de Campos and Huete [20] includes two main steps: detecting the skeleton and assigning the head-to-head connections. One year later, de Campos [21] studied the class of dependency models in singly connected networks using d-separation criterion and developed efficient algorithms (both exact and approximate) for singly connected networks learning from data. To reduce the computational complexity of structure learning, de Campos and Huete [22] proposed the use of conditional independence tests of order zero and one (the cardinality of a conditional independence test) to detect an initial structure and then some additional higher-order tests for network refinement. The conditional independence tests are conducted in terms of the *Kullback-Leibler* cross-entropy. In this approach, it is assumed that the perfect ordering of the nodes is known. Cheng et al. [23] employed an information theoretic dependency analysis method to assess the conditional independence relationships. They developed two algorithms, one for cases with given node ordering information and the other without such information. The algorithms consist of three phases called drafting, thickening and thinning: drafting builds an initial network based on mutual information for node pairs; thickening creates an I-map of the dependency model; finally thinning removes spurious edges using conditional independence tests.

There are two disadvantages of the methods in this category: exponential execution times, and proneness to errors in dependence tests used.

2.2.2 Search-and-score based methods

Among these structure learning methods, search-and-score method is the most commonly used strategy. It identifies a DAG that fits the data better in terms of a pre-defined criterion. As the name

suggested, a search-and-score method typically requires a scoring function and a search strategy. The scoring function is used to assess how well a network fits with the given data and the search method is employed to identify the highest scoring structure over a typically large search space of possible network structures.

Two of the most obvious choices for scoring functions are likelihood score and Bayesian score. Likelihood scoring functions measure $P(D|S)$, the probability of the observed data D given a structure S, i.e., how likely we will observe the training data given a network structure. Bayesian scoring functions, on the other hand, test $P(S|D)$, the posterior probability of the structure given the observed data, by introducing a prior distribution $P(S)$ over the possible values of each network parameters in a likelihood estimation. Note that a prior probability reflects the purely subjective assessment of a network and the posterior probability $P(S|D)$ is proportional to the product of the prior $P(S)$ and the likelihood $P(D|S)$. Cooper and Herskovits [24] described a posterior probability-based scoring function with the assumption of uniform-distributed density function for the probability of parameter vectors given a structure. Interestingly, as pointed out by Bouckaert [25], the Bayesian score developed in [24] may not be the same for two networks with the same set of independency statements. Enlightened by the applications from coding theory, a new measure function based on the *minimum description length* (MDL) was also proposed [25–27] . The MDL measure introduced a penalty term, which induced the principle of Occams razor by penalizing network structures with more edges. Similar scores including AIC criterion and BIC criterion were also introduced for structure learning of Bayesian networks [28]. A potential problem with these scoring methods is that there might exist many network structures with the same score, which makes the search difficult. To address this problem, Heckerman and Geiger [29] introduced some practical methods to assign likelihoods and parameter priors for structure learning. Particularly, they described a new prior called the *Dirichlet distribution* for the parameter variables and derived a score metric called the likelihood equivalence Bayesian Dirichlet score (BDe) [30]. Another issue with score-based structure learning is that many structures may fit the given data equally well. Friedman and Koller proposed an approach called model averaging to address this

problem [31].

Search problem is also referred to as the identification problem, which attempts to identify one or more DAGs with the highest value for a pre-selected scoring criterion. Since the number of possible structures increases exponentially as the number of nodes grows, an exhaustive (*brute force*) search of all network structures is not computationally feasible even for networks of moderate size. For example, for three nodes, the number of possible structures is 25; for ten nodes, this number grows to 4.2×10^{18} [24]. Indeed, Chickering and his fellow researchers [32, 33] have shown that learning Bayesian networks from data is *NP-hard*, even when each node has at most two parents. To reduce the time complexity for structure learning, heuristic search methods are often used. For example, Cooper and Herskovits [24] developed a greedy-search based method, called *K2* algorithm, which assumed that the node ordering information is available. It started from an empty set of nodes and added nodes incrementally according to the probability of the resulting structures. Madigan and York [34] introduced Markov Chain Monte Carlo (*MCMC*) simulation method for structure search. Other search methods, such as genetic algorithms [35, 36], simulated annealing [37], tabu search [38] , and branch and bound [39] were also introduced for network search.

2.2.3 Hybrid methods

Hybrid methods [40–46] combine constraint and search-and-score methods. In general, independence tests are used to build an initial network in order to reduce the search space and search-and-score methods are then applied to the initial networks to detect the DAG that optimizes the score function. Singh and Valtorta used conditional-independence tests to detect the ordering on the nodes and then a Bayesian score to recover the network structures [40, 41]. Spirtes and Meek [42] used the PC algorithm [18] to construct an initial network structure and then a greedy Bayesian search algorithm to construct the final networks. Dash and Druzdzel [43] proposed to search the space of equivalent classes of structures using constraint-based approaches and then to score with Bayesian methods. Acid and de Campos [44, 45] developed a hybrid method that emphasized the

balance between model complexity and accuracy. Most recently, Tsamardinos et al. [46] developed a new algorithm, called *Max-Min Hill Climbing*, which utilized methods in local learning and constraint-based to construct an undirected network, and search-and-score based methods to assign directions.

2.3 Bioinformatics Applications

Structure learning of Bayesian networks has been applied in many fields in computational biology. In some cases, the ultimate goal is to learn the network structures [11, 47–59]. In other cases, structure learning is a necessary step for inference [60, 61] .

In regulatory network reconstruction, we are given time-series *microarray* gene expression measurements for tens of thousands of genes. Our goal is to reconstruct the regulatory networks from these high-throughput expression data. The assumption underlying computational reconstruction of biological networks from microarray data is that the expression levels of a gene are closely tied to those of its transcription factors, which are products of some other genes. In reconstructing regulatory networks, each gene is modeled as a node (variable) and the vector of its expression measurements is modeled as true gene expressions mixed with noise in data generation, variability of the underlying biological systems etc. With this setting, we use the structure learning methods to identify a network structure that maximizes a pre-determined scoring function. We can use the original expression measures, which are continuous, or discretized expression levels (e.g., up-regulated, no change, and down-regulated) [11, 47–57].

Care should be taken when one interprets the reconstructed structures. First, the structure of a Bayesian network is just a simplified version of the biological system as a high-level abstraction. It reflects the roles and interactions of different genes. Second, a Bayesian network may not be able to distinguish the causal relationship between genes. This is because for some cases, two networks of the same architecture may generate the same scores even though some of the edge directions in those networks are different. Possible solutions are to use biologically interventional expression data (e.g., *gene knockouts*) and to integrate biological knowledge (e.g., *transcriptional factors*) into

the learning process.

In discovering protein signaling pathways, we are modeling *intracellular multicolor flow cytometry* data, which are simultaneous observation of multiple phosphorylated protein and phospholipid components in several thousands of human immune system cells [58]. In this application, nodes are molecules in signaling pathways and edges represent the dependencies between these molecules. A Bayesian network learning method is applied to search for the structure that represents a model average from 500 results of high scores. In order to detect the causal influences in cellular signaling networks, the cells are perturbed with a series of stimulatory cues and inhibitory interventions.

Structure learning in Bayesian networks can be applied to many other problems in addition to regulatory networks and signaling pathways. For example, in genome wide association studies, structure learning can be used to model association of hundreds of thousands of single *nucleotide polymorphisms* with a particular disease or phenotype using genotyped data [59]. In addition, BN methods have been applied for inference tasks in bioinformatics [60,61]. The rapid growth in high-throughput technologies and decreasing price in high-throughput experiments will likely provide many more large-scale and large-dimensional biological data, which will offer new applications and significant challenging for Bayesian network learning.

2.4 Challenges and the aims of the thesis

Both constraint-based and search-and-score methods have their own advantages and limitations. By constraining the search space in terms of conditional independence tests, constrain-based methods are relatively quick and do not require the node ordering information. However, in order to accurately recover the network structures, constraint-based approaches often need an exponential number of conditional independence tests with a large number of variables; furthermore, an early mistake in individual independence tests may subsequently lead to a wrong network construction [12]. Search-and-score techniques, on the other hand, focus on the evaluation of the network as a whole and generate networks that fit the training data well (in terms of likelihood or posterior

probability). However, due to the computational complexity, most of the search-and-score methods require the node ordering information, which is not readily available in real applications. Hybrid approaches provide a balanced learning that possesses the search power of a constraint-based algorithm while maintains the efficiency of a search-and-score algorithm.

Regardless of which methods we use, the performance (e.g., network accuracy) of the three classes are quite similar if the sample size is sufficiently large. However, when we apply the methods to bioinformatics problems where sample sizes are very limited, we need to carefully design the structure learning methods and select appropriate evaluation functions. Direct use of the current methods may fail to reliably reconstruct the networks, mainly for the following two reasons:

Sample complexity: It has been shown [61] that many learning algorithms will asymptotically converge to the same 'true' model. With small to medium-size samples, however, the accuracy of learning models will decrease significantly. Ideally, for reliable structure reconstruction, the number of training samples should increase exponentially with the number of nodes. Unfortunately this is not true for our applications: for microarray-based gene network reconstructions, there are typically thousands of genes (each of which is a random variable) and only a few hundred data points (instances).

Computational complexity: Due to the exponential number of possible networks, it has been shown that structure learning is a *NP-hard* problem, even when each node has at most two parents [32, 33]. With gene expression data having several thousands of nodes, identifying an optimal structure with respect to a given evaluation criterion is computationally intractable. Thus, one has to use heuristic search methods to find '*suboptimal*' solutions.

The combination of limited sample size (e.g., number of experiments) and high dimensionality (e.g., number of genes) presents a significant challenge for structure learning in biological network reconstruction. With the exponential number of possible structures and limited sample sizes, it is highly likely for search-and-score methods to produce many models (structures) that have the same high score. In cases like this, one randomly selects a structure generated from these models

as the true DAG [31]. As Friedman and Koller pointed out in [31], '*we have no guarantees that these structural features are even likely relative to the set of possible structures*'. In addition, in gene network reconstruction tasks, one node can potentially have many parents and children nodes (e.g., a *transcription factor* (TF) binding proteins can have several TFs as its parent nodes). For these nodes, the conditional independence tests many not be reliable. My thesis work will address these challenging problems by developing effective structure learning methods for limited sizes of samples with high dimensionality. Our targeted application is to reconstruct gene networks using gene expression data.

Another application of BN in bioinformatics is classification. While most of work has been devoted to structure learning, limited research has been done to address the classification tasks using structure learning. Most of the existing work treats classification as a direct result of structure learning. Since classification and structure learning are different tasks with different goals, an optimal prediction performance may not be reachable if we use the criterion function designed for structure learning. In this thesis, we will develop structure learning methods that are for classification, which is applied to the assessment of protein-protein interactions.

3. Gene Regulatory Network Reconstruction Using Dynamic Bayesian Network Methods

3.1 Introduction

The structure and biological functions of a living cell are attributed to complex interactions between the cell's numerous constitutes, such as proteins, DNA and RNA [62, 63]. Thus, there is a critical need to understand how these molecules interact with each other and the dynamics of the interactions. With the advent of high throughput microarray technologies, mRNA expression levels of tens of thousands of genes can now be measured simultaneously [64, 65]. Knowledge of mRNA levels under different time points provides hints about the effect of external stimuli and the expression levels of other genes on a particular gene, which helps understand the underneath regulatory networks or signaling pathways. Analysis of time-series gene expression data has shown great potential in deciphering cellular networks.

Among a variety of modeling methods, Bayesian networks (BNs) have shown particular promise to infer biological pathways from observational time-series gene expression data [48,54,55,66–68]. BNs are especially suitable for learning regulatory networks or biological pathways for the following reasons: (1) the sound probabilistic semantics allows BNs to deal with the noises that are inherent in experimental measurements; (2) BNs can handle missing data and permit the incomplete knowledge about the biological system; and (3) BNs are capable of integrating the prior biological knowledge into the system.

While static BNs have been successfully applied to reconstruct gene networks from microarray data, BNs have their limitations in that they cannot construct cyclic networks and feedback loops. Since a real gene regulatory mechanism has cyclic regulation including feedback loops, a dynamic Bayesian network (DBN) is preferable to model complex biological networks with feedback loops.

DBNs are capable of capturing cyclic information and dynamic characteristics of regulatory pathways of a cell. Friedman *et. al.* [69] and Murphy and Mian [70] first applied DBN for time-series gene expression data modeling. Ong *et. al.* [71] modeled regulatory pathways in *E. coil* using DBNs and biological knowledge. Perrin *et. al.* [72] used a DBN to test against experimental data relative to the *S.O.S. DNA repair* network of the *Escherichia coli bacterium*. Parameters and missing data were handled via a penalized likelihood maximization approach. Kim *et. al.* [73] proposed a nonlinear regression based DBN model for constructing a gene network from continuous time series gene expression data.

Successful studies on DBNs in biological pathway reconstruction have been reported, two major problems associated with the current DBN methods may limit their applications in large-scale network analysis: relatively low accuracy of regulatory network prediction and excessive computation time [56]. Zou and Conzen [56] proposed to alleviate the two problems associated with standard DBNs by selecting a set of potential regulators for each gene first. The selection of regulators was based on the time points of the initial expression change (*up-regulation* or *down-regulation*) for each gene, which was heuristic in nature: for each gene, the genes that have earlier *up-regulation* would be considered as potential regulators. The proposed method was tested on yeast *cell cycle* data and predicted genetic networks with significantly improved accuracy and reduced computational time compared to the standard DBN methods. However, cautions have to be paid to the selection of appropriate cutoffs for defining *up-regulation* and *down-regulation*. Furthermore, the method would miss some potential genes that were initially *down-regulated* but together with some other initially *up-regulated* genes, also served as other genes' regulators. While the approach was much faster than standard DB approach, the specificity was low, especially for the cases of no prior knowledge of transcript factors.

In this study, we present a novel DBN-based approach for reconstructing the underlying gene networks from both simulated data and real time-series gene expression data. The proposed method uses *differential mutual information* to select potential parents for each node. Furthermore, we propose to adjust *Dirichlet* prior in a Bayesian scoring metric for network regulation in small

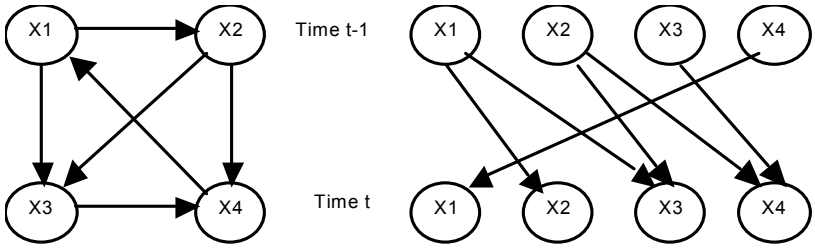


Figure 3.1 – An Example of a dynamic network with cyclic regulations. (a) The directed cyclic graph representation. (b) The directed acyclic graph representation.

sample learning problems. This score is applied to each node to evaluate its parents. The proposed method is first evaluated for recovering two complex networks from simulated data. It is then applied to real yeast time-series gene expression data for reconstructing a *cell cycle* pathway.

3.2 Structure Learning Methods

A dynamic Bayesian network (DBN) is a directed graphic model used for modeling time-series data. It is an extension of BN to model temporal processes. The arcs between the nodes in a DBN are parameterized by conditional probability distributions that model temporal dependencies between variables. Figure 3.1 shows an example of a discrete DBN with four variables, where Figure 3.1(a) is the directed cyclic graph representation of the network, which cannot be learnt by a Bayesian network; Figure 3.1(b) is the corresponding directed acyclic graph representation at two time points $t-1$ and t. A DBN can be used to model a genetic regulatory network from time-series gene expression data: each node represents a gene and the arcs between two nodes indicate regulatory interactions between genes. Thus, reconstructing genetic networks is to learn network structures from experimental data. In order to learn the network structures, we need to construct a scoring metric for model evaluation. We choose to learn the network that maximizes the posterior probability of the network, which is described next.

3.2.1 Bayesian Scoring Metric

Consider a dataset of T samples, $D = (X[1], X[2], \cdots, X[T])^T$, where the t^{th} sample is $X[t] = (X_1[t], X_2[t], \cdots, X_n[t])^T$ and n is the number of nodes (vertices) in a DBN (the superscript T represents transpose). In microarray data analysis, T is the total number of time points (or the number of microarrays), n is the number of genes, and $X[t]$ is the gene expression level vector measured at time t. Fully characterizing a dynamic network requires a joint probability distribution over all the random variables, which could be extremely complex. To reduce model complexity, two assumptions are typically made in DBN [69]: (1) *Markovian* assumption, *i.e.*, $P(X[t+1]|X[t], \cdots, X[0]) = P(X[t+1]|X[t])$; and (2) stationary assumption, *i.e.*, $P(X[t+1]|X[t])$ is a constant, i.e., independent of t. Given these assumptions, the joint probability can be written as

$$P(D|G,\Theta) = \prod_{k=1}^{n}\prod_{t=2}^{T}\theta_k(t) \tag{3.1}$$

where G is the directed network and Θ represents the set of parameters that quantifies the network with components of conditional probabilities $\theta_k(t) = P(X_k(t)|X^{Pa}(t-1))$, $X^{Pa}(t-1)$ is the parent set of $X_k(t)$ in G, which contains nodes at $t-1$. Note that we are interested in the transition network and accordingly the probability of the prior network (at $t = 1$) is ignored.

Using the Bayes rule, the posterior distribution is $P(G|D) = P(G,D)/P(D)$. Since $P(D)$ is independent of the graph G, maximizing the posterior distribution is equivalent to maximize $P(G,D) = P(G) \cdot P(D|G)$, where the likelihood of the data D given a network structure G can be expressed as

$$P(D|G) = \int P(D|G,\Theta)P(\Theta|G)d\Theta \tag{3.2}$$

Under the assumption that the prior distribution over parameters Θ follows the *Dirichlet* distribution with hyperparameters $\alpha_{ijk} > 0$ for $1 \le i \le n$, $1 \le j \le q_i$ and $1 \le k \le r_i$, where q_i is the number of states the parents of node i as a group can take and r_i is the number of states node i

can take [69], the likelihood $P(D|G)$ can be solved in a closed-form [24]. Furthermore, like many others, we assume that no prior knowledge exists about the structure G. Thus, a non-informative uniform prior is used for $P(G)$ (we will drop this term thereafter). An interesting and useful property of the scoring function is that the posterior probability score $S(G:D)$ of the network structure G given the data D is decomposable [24]. As a result, instead of computing $S(G:D)$ for the whole structure, we may evaluate each nodes (and their parents) independently by the following scoring function:

$$S_i(X_i^{Pa}(t-1):D) = \prod_{j=1}^{q_i} \frac{\Gamma(\sum_{k=1}^{r_i} \alpha_{ijk})}{\Gamma(\sum_{k=1}^{r_i}(\alpha_{ijk}+N_{ijk}))} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk}+N_{ijk})}{\Gamma(\alpha_{ijk})} \tag{3.3}$$

where N_{ijk} is the number of times we observe in the data that $X_i(t) = k$ and $X_i^{Pa}(t-1) = s_j$, and Γ is the gamma function.

In microarray-based genetic regulatory network reconstruction, one of the challenging problems is the small sample size leading to poor prediction performance. We herein propose to adjust the hyper-parameters α_{ijk} in order to alleviate the small sample problems and to improve prediction accuracy. While there are many ways to specify α_{ijk}, we investigate a special case where α_{ijk} are set to be a constant. Note that the commonly-used *BDe* (Bayesian *Dirichlet* equivalence) score can be considered as a special case with $\alpha_{ijk} = N/(r_i q_i)$ [30].

The optimal graph can be obtained by maximizing the score in Eq. 3.3 for each node independently. For each node, however, the number of possible sets of parents is quite large. Consequently, searching for the optimal structure is too computationally expensive. Next, we introduce the *differential mutual information* to limit the possible of sets of parents for each node.

3.2.2 Differential Mutual Information

Mutual information (*MI*) has been applied in BN-based structure learning [48, 67]. In this section, we propose the differential MI (*DMI*) for use in DBNs. The *MI* between two random variables X_i and X_j, denoted by $I(X_i, X_j)$, is defined as the amount of information shared between the two

variables. It is used to detect general dependencies in data. $I(X_i, X_j)$ is mathematically defined as follows:

$$I(X_i, X_j) = H(X_i) - H(X_i|X_j) \tag{3.4}$$

where, $H(X_i)$ represents the entropy of the random variable X_i and $H(X_i|X_j)$ represents the conditional entropy of random variable X_i given the random variable X_j [74]. Entropy and Conditional Entropy are mathematically defined by the following equations:

$$H(X_i) = -\sum_{k=1}^{r_i} P(X_i = k) \cdot \log(P(X_i + k)) \tag{3.5}$$

and

$$H(X_i|X_j) = -\sum_{k=1}^{r_i}\sum_{l=1}^{r_j} P(X_i = k, X_j = l) \cdot \log(P(X_i = k|X_j = l)) \tag{3.6}$$

Here, $P(X_i = k, X_j = l)$ is the joint probability distribution of the two random variables X_i and X_j, and $P(X_i = k|X_j = l)$ is the conditional probability of the random variable X_i given the random variable X_j.

In a DBN, all the parents (potential regulators) $\{X_j\}$ of a node (gene) X_i at time t come from nodes at time $t-1$. Thus, for each node X_i, in order to find its potential parents $\{X_j\}$, we need to consider the mutual information between the node X_i to all the other nodes (including itself) in previous time slot. This can be achieved by calculating the joint probability in Eq. 3.6 as follows (the conditional probability can be computed based on the joint probability):

$$P(X_i = k, X_j = l) = \sum_{t=1}^{T-1} \delta(X_i(t+1), X_j(t))/T \tag{3.7}$$

where, $\delta(X_i(t+1), X_j(t)) = 1$, if $X_i(t+1) = k$ and $X_j(t) = l$; $\delta(X_i(t+1), X_j(t)) = 0$, otherwise.

Since this *MI* is based on nodes from different time, we call it differential mutual information (*DMI*). *DMI* measures the dependency between two random variables. The greater the *DMI* values between two random variables, the more closely they are related. If there is a direct edge (con-

nection) between two nodes X_i and X_j, there exists a strong dependency between these two nodes. Thus, we can identify a set of potential parents for each node based on the *DMI* values between this node and other nodes. Instead of considering all the nodes as potential parents which is too computationally expensive (exhaustive search methods) or applying some heuristic rules such as hill-climbing, which may find local optimum, we identify a small set of potential parents, and run exhaustive search on the reduced set of parents. As the experimental results demonstrate in next section, the proposed method can improve both prediction accuracy and computational efficiency.

3.3 Experimental Results

We first use simulated data drawn from known structures to evaluate the proposed method (Section 3.3.1), which is compared with the commonly-used *BDe* score, Maximum Likelihood (*ML*) score, and Bayesian information criterion (*BIC*) score. Our method is then applied to real expression data to recover the cell cycle pathway in yeast (Section 3.3.2).

3.3.1 Experiment on simulated data

Data collection and network evaluation

To evaluate the proposed method, we simulate two randomly generated networks: one with 10 nodes and 11 edges (Fig. 3.2-(a)), and the other with 15 nodes and 21 edges (Fig. 3.2-(b)). Both networks are cyclic with feedback loops. The conditional probability tables are randomly generated with three states for each node. The existence of the known network structures allows us to define two important terms, which indicate the performance of the algorithm:

1. *missing edges* - edges present in the original network but not in the learnt network structure

2. *wrong edges* - either edges not present in the original network but included in the learnt network structure or edges present in the learnt network structure, but having opposite orientation when compared to the corresponding edge in the original network structure.

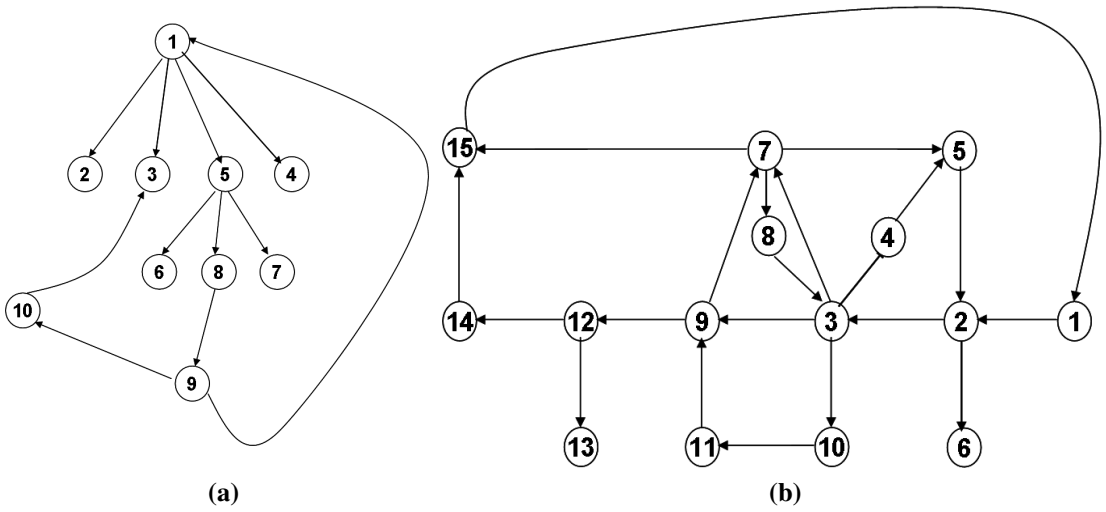
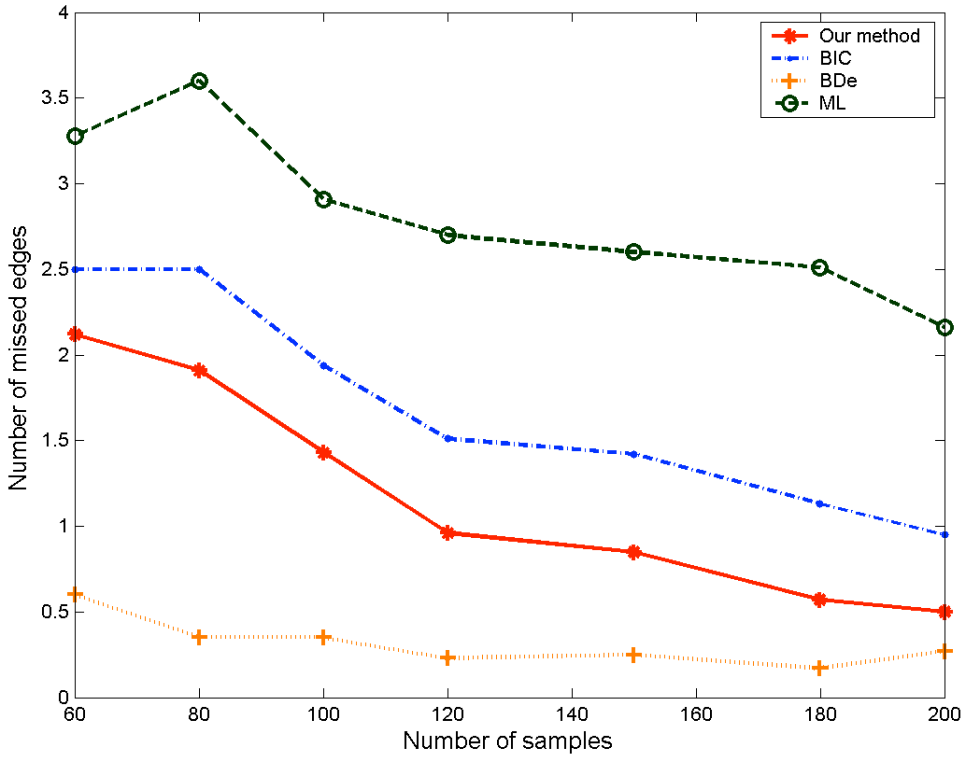


Figure 3.2 – Two cyclic networks: (a) A network of 10 nodes and 11 edges; (b) A network of 15 nodes and 21 edges

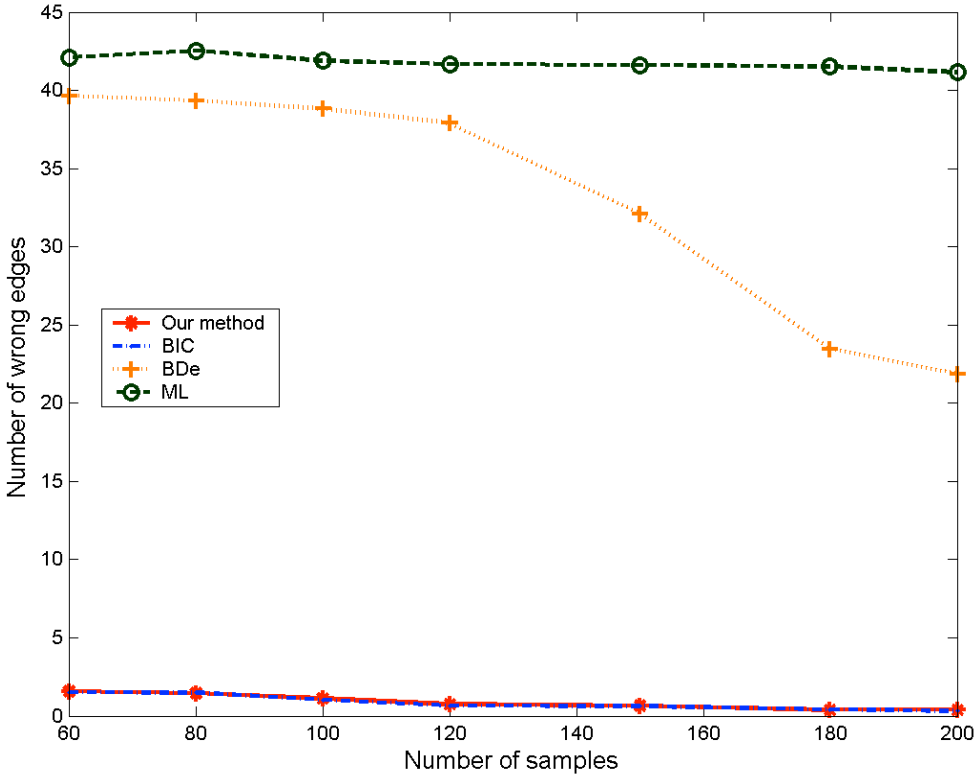
Experimental results

We use existing DBN software (*Bayes Net Toolbox*, downloaded from http://bnt.sourceforge.net/) for *BIC* and *ML* methods. The parameter maximum *fan-in* (the maximum allowable number of parents) in all the methods is chosen as 5. For each network, 50 independent datasets are generated. The reconstructing results are averaged over 50 datasets for each network. The number of samples generated from each network ranges from 60 to 200, which are typical sample sizes available in microarray datasets. In our experiment, the larger the hyper-parameters are, the more true edges (true positives) are predicted correctly; however, the more false edges (false positives) are introduced as well. To reduce false positives, a small value for the hyper-parameters is preferable. One may determine the hyper-parameters through cross-validation methods. In this study, for all the networks, we choose $\alpha_{ijk} = 0.96$. We observed that for $\alpha_{ijk} \in [0.9, 1]$, the prediction results are quite similar.

Figure 3.3 and Figure 3.4 show the number of missing and wrong edges for the two networks predicted with different number of samples. Both *BDe* and our method miss less number of edges than *BIC* and *ML* methods (Figure 3.3(a) and Figure 3.4(a)). While it can identify the largest number of true edges, *BDe* method introduces a much larger number of false positives compared

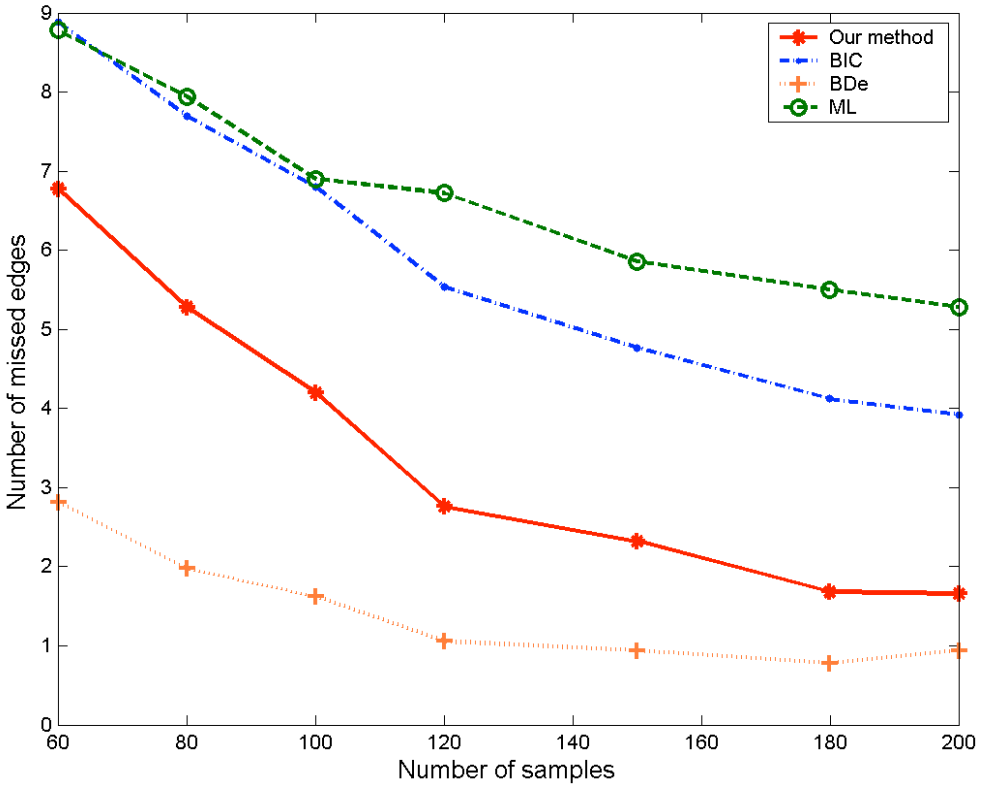


(a)

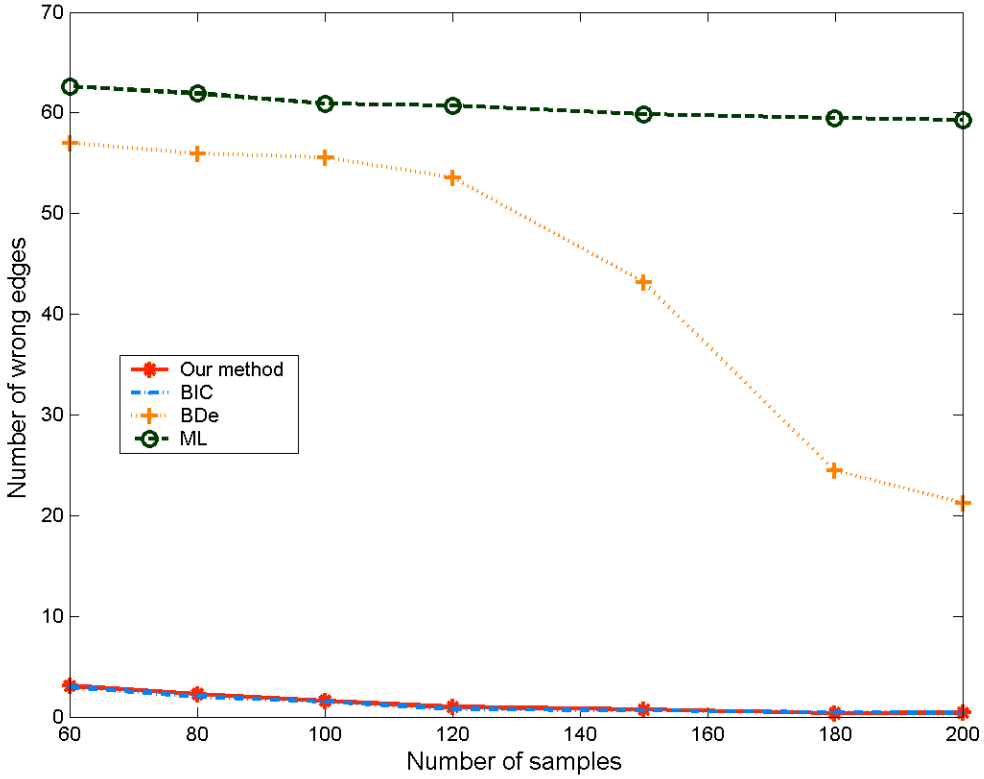


(b)

Figure 3.3 – Experimental results: (a) Number of missing edges; and (b) number of wrong edges versus number of samples for the network of 10 nodes.



(a)



(b)

Figure 3.4 – Experimental results:(a) Number of missing edges; and (b) number of wrong edges versus number of samples for the network of 15 nodes.

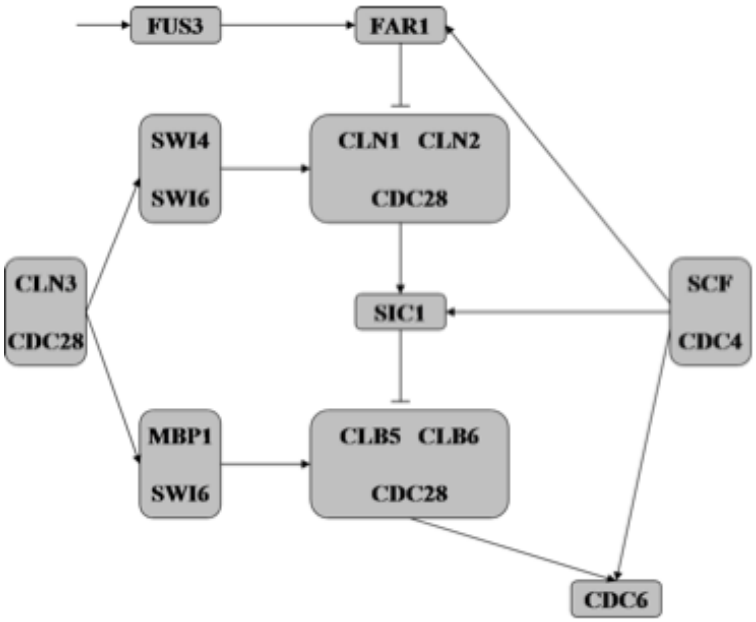


Figure 3.5 – Cell cycle pathway. Squares represent complex with more than one gene

to our method (Figure 3.3(b) and Figure 3.4(b)). Overall, our method produces the best results in terms of the prediction accuracy (both false positive and true positive). Furthermore, our method is much faster than *BIC* and *ML* methods. For example, with 200 samples, for the network of 10 nodes, our method needs 3.9s and *BIC* and *ML* methods need 9s. The computational time for *BIC* and *ML* methods increases significantly as the number of nodes increase. For the network of 15 nodes, our method needs only 8s and *BIC* and *ML* methods need 114s. Note that *BDe* method requires same amount of time as our method needs, as *BDe* is a special case of the proposed methods.

3.3.2 Experiment on Real Expression Data

We apply our method to analyze *S.cerevisiae cell cycle* time-series expression data [75]. A 3-level quantization scheme is chosen to discretize the continuous expression data. We consider a cell cycle pathway that is extracted from the *KEGG* database (www.genome.jp/kegg). The cell cycle pathway consists of 15 genes involved in *G1* phase and *S* phase (Figure 3.5).

For the cell cycle pathway, our method identifies 14 edges: nine of them are correct (Table 3.1).

Table 3.1 – Identified gene-gene interactions in cell cycle pathway

Corrected identified gene-gene interactions				
FAR1-CLN1	CLN1-CLN2	CLN1-SWI6	SWI4-SWI6	MBP1-SWI6
CLB5-CLB6	CDC28-CDC6	CDC4-SCF	FAR1-CDC4	
Identified gene-gene interactions not shown as direct edges in the original network				
FUS3-SCF	CLN2-CLB6	CLN2-CLN3	SIC1-FAR1	CLN3-CDC6

Among the five incorrect ones, four edges (interactions) are existent through a single intermediate node (or genes). Indeed, other experiments demonstrate that some of the predicted gene pairs that are not shown in the KEGG pathway do form interactions with each other in cell cycle pathway. For example, Basco *et al.* [76] conducted biological experiments and showed that *CLB6* interacted with *CLN2*. They observed that (1) the acceleration of the G1-to-S phase transition conferred by *CLB6* was mediated via interactions with *CLN2*, and (2) *CLB6* is responsible for down-regulation of the protein kinase activities associated with *CLN2* during cell cycle progression in vivo. Similarly, Wijnen and Futcher [77] demonstrated biologically that *CLN 3* activities transcription of *CLN2* during the G1-S transition in the cell cycle of *Saccharomyces cerevisiae*.

3.4 Conclusions

A genetic regulatory network often contains feedback loops for further regulation of genes. To characterize the cyclic nature with feedback loops, DBNs have been extensively applied for learning regulatory networks from gene expression data. In this study, we propose a novel method to address two limitations in standard DBN methods, namely, low accuracy of prediction and extremely large computational time. This method is tested on two simulated networks and a real biological pathway. Our experimental results show that the proposed method produces better overall performance than the commonly-used *BDe*, *ML*, and *BIC* methods. Furthermore, our method is much faster than *BIC* and *ML* methods, especially for networks of moderate size. This is attributed to the fact that we only evaluate a limited number of potential parents (selected by our *DMI*) for each

node. Thus, our method can be applied to very large networks learning where standard DBN is infeasible. Experimental results carried out on the cell cycle pathway of *Saccharomyces cerevisiae* further demonstrate the effectiveness of the proposed method. We predicted some new interactions that were not included in the original *KEGG* pathways, but verified by biological experiments elsewhere.

4. Markov Blanket Based Structure Learning Method and its Applications

4.1 Introduction

The Markov blanket , $Mb(X)$, of a random variable X in a variable set (V) (or its corresponding node in DAG) is the smallest set of variables, such that $X \perp \{V - X - Mb(X)\}|Mb(X)$; or, from a DAG point of view, $Mb(X)$ is the set of its parents, children and spouses (children's parents) [15] (refer to Chapter 2, page 3). The concept of Markov blankets reflects a fundamental property of a Bayesian network. Explicitly using Markov blankets in the Bayesian structure learning can effectively limit unnecessary computation, since the Markov blanket of a node contains great information of its local structure [78].

As we know, in order to accurately recover the network structures, constraint-based approaches often need an exponential number of conditional independence tests with a large number of variables; furthermore, an early mistake in individual independence tests may subsequently lead to a wrong network construction [12]. Many structure learning methods are restricted from large size network applications because of their exponential execution times and proneness to errors in dependence tests. However, when the knowledge of Markov blankets of the nodes is available, we only need to focus on limited number of close nodes , which makes structure discovery much faster and more reliable.

The above definition of Markov blanket also suggests a way to identify it: Markov blanket of a variable is a strong relevant set of the variable. Thus, one can find Markov blanket, using many existing feature selection methods. Tsamardinos and Aliferis [79] showed that under faithful distributions, the Markov blanket of a variable is exactly the set of strongly relevant features, and proved its uniqueness. Pellet and Elisseeff [80] also theoretically analyzed the equavilence between

relevance sets and Markov blankets. They proved under the assumption that feature selection returns the Markov blankets of the variables, the approximate *moral graph* based on the Markov blankets can be adjusted locally into correct PDAG (partially directed acyclic graph) depicting the causal structure. The moral graph is the counterpart of a directed acyclic graph, formed by connecting nodes that have a common child, and then making all edges in the graph undirected. Equivalently, a moral graph of a directed acyclic graph G is an undirected graph in which each node of the original G is now connected to its Markov blanket [81, 82].

In this work, we first rely on feature selection to return Markov blankets of each node in a Bayesian network, then exam the dependence relationships in the neighborhood to further determine the local structures which satisfy the dependence constrains.

4.2 Method

This section is organized as follows. In Section 4.2.1, we analyze the likeness between the outcome of feature selection and Markov blanket, then review a feature selection algorithm we use in this study. In Section 4.2.3, we show the correctness and efficiency of *Collider Set* algorithm proposed by Pellet *et.al.* [80], an algorithm we use in local structure recovery. In Section 4.2.4, we describe our algorithm.

4.2.1 Feature selection and Markov blanket construction

Feature selection

Feature selection (*FS*) is an important step in many machine learning and statistics tasks. It removes redundant and irrelevant features to reduce computational complexity and build robust learning models. In classification, ideally, FS returns a set of *strongly relevant* features related to the class variable (target). Strong relevance is defined as:

Definition 1 (Strong relevance) [83]: In a variable set V, a variable $X \in V$ is strongly relevant to a target $Y \in V$, if $P(Y|\{V - X\}) \neq P(Y|V)$. In other words, a variable is strongly relevant to

a target if it carries information about the target that no other variable provide. According to the definition of conditional independence, we have, $X \not\perp Y|\{V-X-Y\}$ (refer to Chapter 2, page 3), i.e., given all the other variables, X and Y are still conditionally dependent.

If we denote the set of strongly relevant features of target Y as F_Y, then

$$F_Y = \{X|(X \not\perp Y|\{V-X-Y\}\} \tag{4.1}$$

In other words, F_Y is the set of the variables that are dependent on the target Y, conditioned on all others. Based on the symmetry of the conditional independence, we have, $X \in F_Y \Leftrightarrow Y \in F_X$, where F_X is the strongly relevant set of X.

Markov blanket

Property of Markov Blanket (Total conditioning) [80]: In the context of a faithful causal network ($\mathscr{G}$), we have:

$$\forall X,Y \in V : (X \in Mb(Y) \Leftrightarrow (X \not\perp Y|\{V-X-Y\}) \tag{4.2}$$

Namely, the Markov blanket of a node is the set of all variables that are dependent on the node conditioned on all other variables, which means Markov blanket contains all the information relevant to the node. In this thesis, Markov blanket also refers to Markov boundary, i.e., no redundant variables are allowed in a MB. Summarily, the Markov blanket of a target, Y, contains all relevance and no redundancy, i.e., it has the same property as the strongly relevance set of Y. Thus, we draw the conclusion that strongly relevance set of a variable is equivalent to its Markov blanket:

$$F_Y \equiv Mb(Y) \tag{4.3}$$

4.2.2 Markov blanket identification algorithm

In this work, we apply an efficient feature selection algorithm proposed by Margarits and Thrun [78] to identify the Markov blanket of each node in the graph. The algorithm is summarized in

Table 4.1 – The Grow and Shrink algorithm to indentify Markov Blankets

Compute Markov Blankets : Let $X \in V$, compute the Markov blanket $Mb(X)$

1. $S \leftarrow \phi$
2. While $\exists Y \in V - \{X\}$ such that $Y \not\perp X|S$, do $S \leftarrow S \cup \{Y\}$. Growing phase
3. While $\exists Y \in S$ such that $Y \perp X|S - \{Y\}$, do $S \leftarrow S - \{Y\}$. Shrinking phase
4. $Mb(X) \leftarrow S$

Table 4.1. It consists *grow* and *shrink* two phrases. Initially, let the relevant set S of X be empty. In the growing phrase, as long as the Markov blanket property of X is violated, that is, there exists a variable $Y \in V$ dependent on X, we add Y to the current set S until there are no more such variables. In this process, however, there may be some variables that were added to S that were really outside the blanket. Such variables would have been rendered independent from X at a later point when 'intervening' nodes of the underlying Bayesian net were added to S. This observation necessitates shrink step, which identifies and removes those variables. The algorithm is efficient, requiring only $O(n)$ conditional tests, making its running time $O(n|D|)$, where $n = |V|$ and D is the set of samples.

4.2.3 Construction of Causal graphs using feature selection

It is a common practice to use Bayesian network to represent causal relationships, however a directed edge from X to Y doesn't require that Y is causally dependent on X. This can be explained by the fact that $X \rightarrow Y \rightarrow Z$ and $X \leftarrow Y \leftarrow Z$ are equivalent, i.e., they imply the same conditional independence requirements.

A *causal network* is a Bayesian network with an explicit requirement that the relationships be causal. The additional semantics of the causal networks specify that if a node X is actively caused to be in a given state x (an action written as $do(X = x)$), then the probability density function

changes to the one of the network obtained by cutting the links from X's parents to X, and setting X to the caused value x [14]. Using these semantics, one can predict the impact of external interventions from data obtained prior to intervention. In the rest of this Chapter, we assume that the Bayesian networks being learnt are causal networks.

If we make the *Causal Sufficiency* assumption [84], that is, assume that no hidden common cause of two variables exists, we can write:

$$\forall S \subseteq (V - \{X,Y\}) : (X \not\perp Y|S) \Rightarrow (X \rightarrow Y \text{ or } Y \rightarrow X) \tag{4.4}$$

Using Equation 4.4, we can theoretically determine all adjacencies of the causal network with conditional independence tests, but we cannot orient the edges. The only causational pattern where conditional-independence tests can reveal the direction of causation is *V-structure*. This is known as *causal under determination* (refer to Spirtes' book [84], p.62), i.e., for the structure learning task given observational data, a correct graph is specified by its adjacencies and its V-structures only. *Partially oriented DAG* (PDAG) returned by structure learning algorithms represent observationally equivalent classes of causal networks (refer to Pearl's book [14], p.19). The fact narrates that for a given joint probability distribution $P(V)$, the set of all conditional independence statements that hold in $P(V)$ does not yield a unique *perfect map* (refer to Chapter 2, page 3) in general.

V-structure [14] consists of two common causes X, Y and one common effect Z (called *collider*). X and Y initially independent, become dependent when conditioned on a Z. To identify V-structure, first, we certify the existence of a link between X and Z and between Y and Z, Z is then identified as an unshielded collider if conditioning on it creates a dependency between X and Y.

The result of feature selection can be graphically shown by an undirected graph $G = \langle V,E \rangle$ where $(X,Y) \in E \Leftrightarrow X \in F_Y$. This graph is close to the original causal graph in that it contains all arcs as undirected links, and additionally links spouses together, and is called the moral graph of the original directed graph (refer to Lauritzen and Spiegelhalter' book [82], p.166). The extra

step needed to transform this graph into a causal PDAG is the deletion of the spouse links and the orientation of the arcs. Pellet and Elisseeff [80] proposed a fast algorithm named *Collider Set algorithm* (CS algorithm) for finding hidden spouse link and V-structure, by identifying collider nodes. Collider set is defined as:

Definition 2 (Collider set) [15]: In an undirected graph $G = \langle V, E \rangle$, the Collider set of a pair of adjacent nodes (X and Y, $(X,Y) \in E$), denoted as $Tri(X-Y)$, is the set of vertices forming a triangle with X and Y :

$$Tri(X-Y) = \{Z \in V | (X,Z) \in E, (Y,Z) \in E\} \tag{4.5}$$

Suppose that G is the moral graph of the DAG representing the causal structure of a faithful data set. A set of vertices $Z \subseteq Tri(X-Y)$ then has the Collider Set property for the pair (X,Y) if it is the largest set that fulfills:

$$\exists S_{XY} \subseteq \{V-X-Y-Z\} : (X \perp Y | S_{XY}) \text{ and } \forall Z_i \in Z : (X \not\perp Y | S_{XY} \cup \{Z_i\}) \tag{4.6}$$

The set S_{XY} is then a d-separating set for X, Y. Pellet and Elisseeff [80] proved that in the content of a faithful causal network, for a given pair (X,Y), the set Z that has the Collider Set property exists if and only if X is neither a direct causal nor a direct effect of Y. This set Z is unique when it exists. In the large sample limit, for faithful, causal sufficient data sets, the CS algorithm correctly identifies all V-structures and all spouse links, assuming consistent statistical tests.

It is sometimes possible to orient further arcs in a graph by looking at already-oriented arcs and propagating constraints, preventing acyclicity and the creation of additional V-structures other than those already detected. The graph after this constraint-propagation step is called complete PDAG.

Table 4.2 – Causal Network Learning Algorithm

1: **procedure** StructureLearning
   Input D : $n \times d$ dataset with n d-dimensional data points
   Output G : maximally oriented partially directed acyclic graph
2: /* **Step 1: Markov blanket construction** */
   for each variable $X \in V$ do
   $F_X \leftarrow$GrowShrinkAlgorithm$(X; D)$
   end for
   for each pair (X, Y) such that $Y \in F_X$ and $X \in F_Y$ do /* symmetry check */
   add X to $Mb(Y)$ and Y to $Mb(X)$
   end for
3: /* **Step 2: Spurious arc removal and V-structure detection** */
   $G \leftarrow$ CSAlgorithm[$Mb(.)$]
4: /* **Step 3: Constraint propagation** */
   $G \leftarrow$ CompletePDAGAlgorithm[G]
5: end procedure

4.2.4 Algorithms

Our method for causal network learning using Markov blankets can be divided into three steps: first, employ Grow and Shrink algorithm [78] (refer to Section 4.1, page 30) to identify Markov blankets for all the nodes in the graph; second, construct morph graph based on Mb(.), and apply Collider Set (CS) algorithm [80] explained in Section 4.2.3 to remove spouse links and orientate V-structure; finally, use the Complete PDAG algorithm [80] to orientate PDAG maximally. The above procedure is illustrated in Table 4.2, and the details of Collider set algorithm and Complete PDAG algorithm are lists in Table 4.3 and Table 4.4 respectively.

4.3 Experiment: RAF pathway learning with small sample datasets

Objective

The experiment is designed to compare our algorithm with two well-known structure-learning algorithms: *Hill-climbing* and *MCMC*. We are especially interested in the comparison of their performance with limited sample sizes, a common situation in processing biological data. We

Table 4.3 – Collider Set Algorithm

```
1:  procedure: CSAlgorithm
        Input: Mb(X) : the Markov blanket information for each node X ∈ V
        Output: G : partially oriented DAG
2:      G ← moral graph according to Mb(X)
3:      C ← {}, an empty list of orientation directives
4:      for each edge X − Y part of a fully connected triangle do
5:          S_XY ← null /* search for d-separating set */
6:          B ← smallest set of {Bd(X)\Tri(X−Y)\{Y}; Bd(Y)\Tri(X−Y)\{X}}
7:          for each S ⊊ Tri(X−Y) do /* subset search */
8:              Z ← B ∪ S
9:              if CondIndep(X;Y|Z)then
10:                 S_XY ← Z
11:                 break to line 23
12:             end if
13:             D ← B∪ {nodes reachable by W in G\XY|W ∈ Tri(X−Y)\S}
14:             B′ ← B\D
15:             for each S′ ⊊ D do /* descendant of collider may be opening a path */
16:                 Z ← B′ ∪ S′ ∪ S
17:                 if CondIndep(X;Y|Z) then
18:                     S_XY ← Z
19:                     break to line 23
20:                 end if
21:             end for
22:         end for
23:         if S_XY ≠ null then /* save orientation directive */
24:             mark link X − Y as spouse link in G
25:             for each Z ∈ (Tri(X−Y)\S_XY)do
26:                 C ← C ∪ {(X → Z ← Y)}
27:             end for
28:         end if
29:     end for
30:     remove all spouse links (i.e., marked links) from G
31:     for each orientation directive (X → Z ← Y) ∪Cdo /* orient edges */
32:         if edges X − Z and Y − Z still exist in G then
33:             orient edges as X → Z ← Y
34:         end if
35:     end for
36:     return G
37: end procedure
```

Table 4.4 – Complete PDGA algorithms

```
1:  procedure CompletPDAGAlgorithm
        Input G : partially directed acyclic graph
        Output G : maximally oriented partially directed acyclic graph
2:      while G is changed by some rule do /* fixed-point iteration */
3:          for each X, Y, Zsuch that X → Y − −Z do
4:              orient as X → Y → Z /* no new V-structure */
5:          end for
6:          for each X, Y s.t. X − −Y and ∃ directed path from X to Y do
7:              orient as X → Y /* preserve acyclicity */
8:          end for
9:          for each X, Y s.t. X − −Y and ∃ nonadj. Z, W s.t. X − −Z → Y and X − −W → Ydo
10:             orient as X → Y/* three-fork V with married parents */
11:         end for
12:     end while
13:     return G
14: end procedure
```

apply *Bootstrapping* technique which is a frequent choice for handling the limit sample issue.

dataset description

The data for our experiments are from both real world and synthetics. The real data from flow cytometry experiments [57], which can simultaneously measure protein expression levels in thousands of individual cells, are used to learn RAF biological regulation pathway. The Ras/Raf/MEK-pathway is at the heart of signalling networks that govern proliferation, differentiation and cell survival [85]. The deregulation of the Raf pathway can lead to carcinogenesis, and the pathway has therefore been extensively studied in the literature(e.g., [57, 85, 86].The RAF pathway includes 11 proteins and 20 causal connections as shown in Figure 4.1.

Sachs *et al* [57] did a series of stimulatory and inhibitory interventions targeting on different proteins in the RAF pathway, and collected nine datasets under different biological conditions. Each dataset contains more than 700 samples (cells), and for each sample, the expression levels of the 11 proteins in the pathway were recorded. Table 4.5 summarizes the nine datasets: two datasets without intervention (called Original) and seven with interventions (called Perturbation).

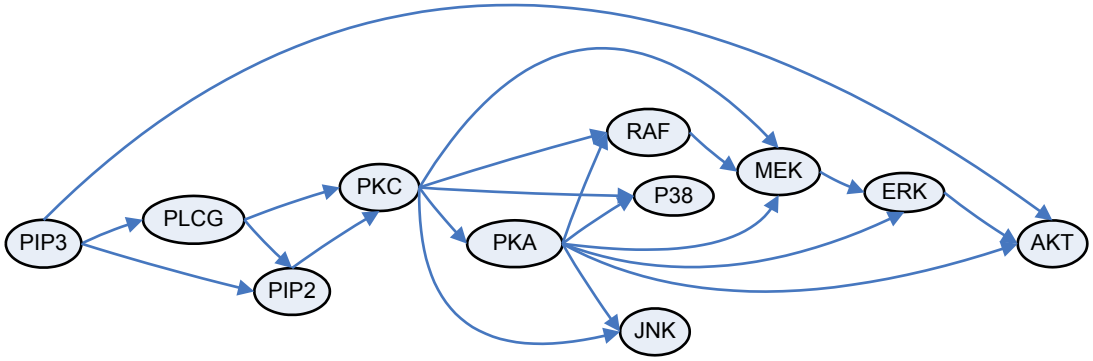


Figure 4.1 – Raf signalling pathway. The graph shows the currently accepted signalling network, taken from [57]. In the interventional studies, the following nodes were targeted. Activations: PKA and PKC. Inhibitions: PIP2, AKT, PKC and MEK.

Table 4.5 – Datasets for RAF pathway experiments: Real datasets, where '-' stands for inhibit and '+' for activate

Dataset	1	2	3	4	5	6	7	8	9
Perturbation	no	no	-AKT	-PKC	-PIP2	-MEK	-AKT	+PKC	+PKA

Based on the RAF pathway and simulating the same biological conditions as the real datasets, we also generate nine synthetic datasets, using software Netbuilder [87]. Therefore, the data can be categorized as Real Original (RO), Real Original and Perturbation (ROP), Synthetic Original (SO, 1400) and Synthetic Original and Perturbation (SOP). All the following experiments related to comparison among the algorithms are executed under each of the four categories.

Data preprocess

For real data, we select same number of samples (700) from each of the nine datasets, then discretilize these 6300 continuous values into 2 levels and normalize them for each variable (protein). Synthetics data are quantized and normalized in the same way.

Data sizes and algorithms performance

First, with all the samples under each data category as training samples, we learn the Bayesian Network (BN) of the pathway, using Hill-climbing, MCMC, and our method. Second, we only randomly select N (sample size) samples without replacement to train the BN. N is set at 54, 108, 162, 216, 270 and 324 in our experiments. Repeat the procedure 100 times for each sample size

and average the performance for each algorithm.

Small datasets with bootstrapping

First, we construct a base dataset for bootstrapping under each data category, by randomly selecting N (e.g., 324) samples without replacement. Second, randomly select 90% of N ($324 \times 90\% \approx 292$) samples *with replacement* from the base dataset to form a training set, then learn the structure and obtain its adjacent matrix A_i. Repeat the previous step 100 times, and calculate the average A_v of the 100 adjacent matrices obtained from each algorithm, transform A_v to a new adjacent matrix A_{adj} by setting the (i, j) element $a_{ij} = 0$ if the (i, j) element in A_v is less than the pre-specified threshold T , otherwise $a_{ij} = 1$. Finally evaluate the performance of each algorithm by comparing the new adjacent matrix A_{adj} with the one of the golden structure. Threshold T is set at 0.5 in our experiments (we tried 0.3 and 0.2 as well in pre-test and the performance is worse).

All the conditional independence tests are based on partial correction and Fisher's Z-transform [88].

4.4 Results and Conclusion

In general, we evaluate the learned structure by comparing it with the golden structure and counting the number of correctly predicted edges, the number of missing edges, the number of extra edges and the number of wrongly oriented edges. However in our case, there are only two V-structures in the golden net (Figure 4.1), which means that we can only determine the orientations of three edges (i.e., $PIP3 \rightarrow AKT, ERK \rightarrow AKT$ and $PKA \rightarrow AKT$) out of the 20 edges by conditional independence test. In other word, most of the wrongly oriented edges compared with the golden structure can be counted as correct predictions in terms of Markov observational equivalence. Based on the above observation, we also use the sum of missing and extra edges (SME) as our evaluation criterion.

In Table 4.6, HC stands for Hill-climbing algorithm, MCMC for MCMC algorithm, MB for MB based algorithm (our method). Table 4.6 shows the learning results when using the complete datasets to train models. Figure 4.2 to Figure 4.11 present the comparison of learning accuracy in

Table 4.6 – Comparison among learned BN graphs with Hillclimbing, MCMC and Markov Blanket methods from different types of RAF datasets: In the table, RO, ROP, SO and SOP stand for original real dataset, real data with perturbation (or intervention), simulated original dataset, simulated dataset with perturbation respectively; HC stands for Hillclimbing search method; MB means Markov Blanket, i.e., our method; SME means sum of missing and extra edges.

Data Category	correct edge			missing edge			extra edge			SME			wrong orientation		
(Num. of samples)	HC	MCMC	MB	HC	MCMC	MB	HC	MCMC	MB	HC	MCMC	MB	HC	MCMC	MB
RO(1400)	3	6	**8**	12	13	12	3	3	6	15	**16**	18	5	1	**0**
ROP(6300)	**8**	7	**8**	5	4	9	19	17	11	24	21	**20**	7	9	**3**
SO(1400)	**7**	4	6	9	8	6	16	14	12	25	22	**18**	**4**	8	8
SOP(6300)	5	10	**17**	2	2	3	14	16	8	16	18	**11**	13	8	**0**

terms of number of correct edges, number of missing edges, number of extra edges, sum of missing and extra edges and number of wrong directed edges among learned BN graphs with Hillclimbing, MCMC and Markov Blanket methods from different types of RAF datasets and different sample sizes. From Figure 4.2 to Figure 4.6, for a certain sample size (e.g. 216), the results are average over100 datasets at the same sample size (e.g., 100 datasets all with 216 samples). From Figure 4.7 to Figure 4.11, for a certain sample size (e.g. 216), the results are averaged over 100 datasets generated by bootstrapping resampling from the same original dataset at the sample size (e.g., 100 datasets generated by randomly selecting $216 \times 90\% \approx 194$ samples with replacement from the same randomly selected base dataset with 216 samples).

In Table 4.6, bold numbers indicate the best results in a category with the same metric and data type. No best numbers are listed in the missing and extra metrics, as we are more interested in the sum of missing and extra edge, which describes the overall and balanced aspect. As Table 4.6 shows, for the two data types with perturbation (ROP and SOP), our method obtains best results in number of correct edge, sum of missing and extra and number of wrongly orientated; especially, for simulated perturbation (SOP), our method produces apparently best performance among all three methods.

In Figure 4.2, as we expect, for all three methods, datasets with perturbation information (ROP and SOP) produce relatively better results than the dataset without those information; as sample size increases form 54 to 324, in all data types and all methods, the number of correct edges

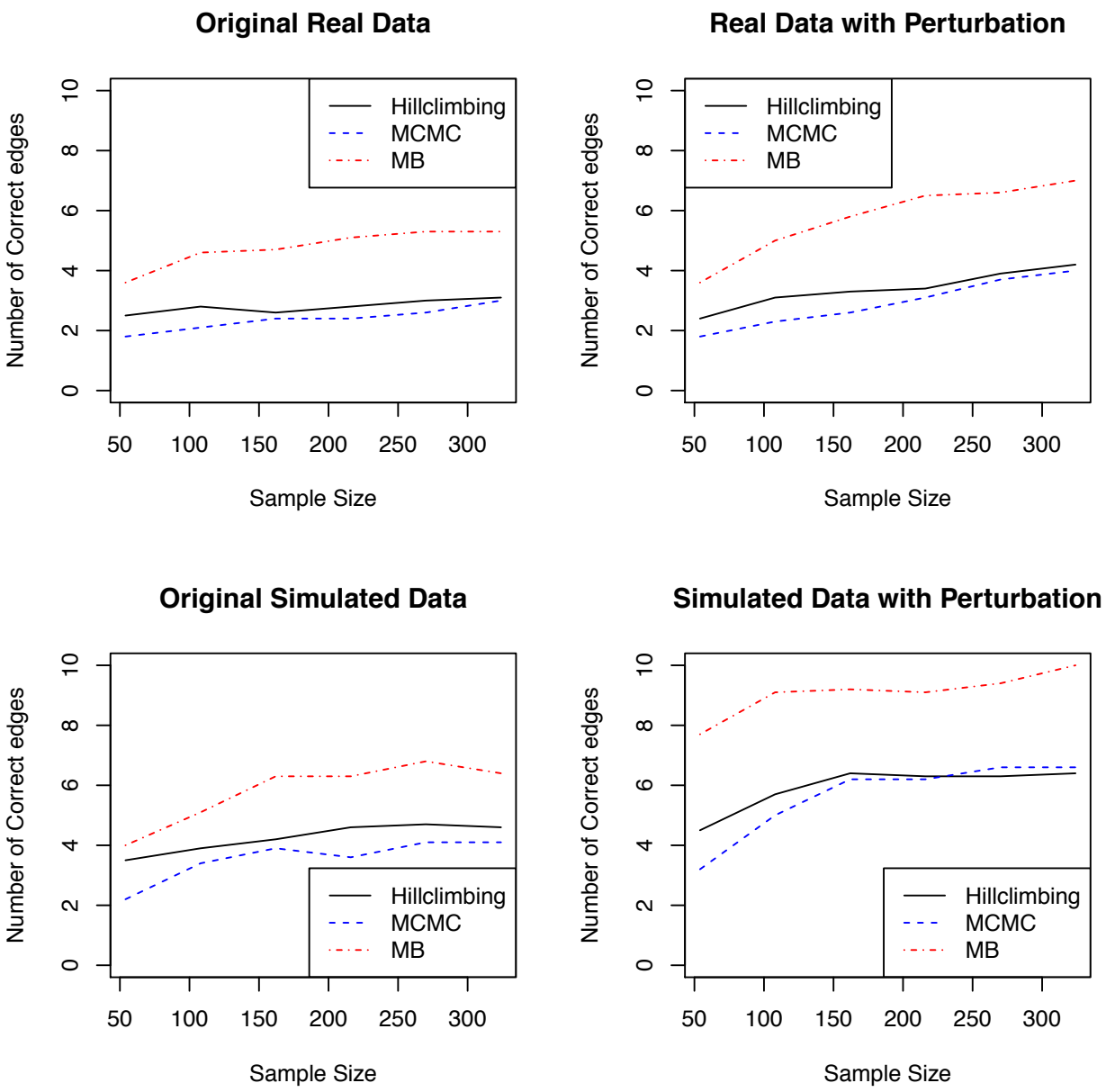


Figure 4.2 – Plot of Sample Size vs Number of Correct edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm

increases. The figure also shows that for all four data types, our method obtains more correct edges than Hillclimbing and MCMC. An interesting observation is: for simulated data with perturbation, our method obtains more correct edges ($\approx$8) at the smallest sample size (54) than the most number of correct edges($\approx$6) at the largest sample size (324) from Hillclimbing and MCMC.

In Figure 4.3, as we expect, for all three methods, datasets with perturbation information(ROP and SOP) produce relatively better results than the dataset without those information; as sample size increases form 54 to 324, in all data types and all methods, the number of missing edges decreases. However, the three methods produce similar results,

In Figure 4.4, we see the number of extra edges increase as the sample size increases, which means false positives increase. Still, our method produces least extra edges in SOP datasets.

In Figure 4.5, all lines are relatively flat, which means the sum of missing and extra edges is not observably effected by sample size, at least in the range of sample sizes we are considering. In spite of that, we still observe that our method perform best among all method for SOP data type.

In Figure 4.6, for all four data types, our method produces clearly best performance (least number of wrong directed edges) among all methods, although as sample size increases, the errors increase in all methods.

In Figure 4.7 to Figure 4.11, which present the results from bootstrapping, we obtain similar observations as the group without bootstrapping, even though the lines are not smooth. The unevenless observed from bootstrapping group is caused by much less information (1/100) than the first group. For instance, the results from sample size 216 in bootstrapping experiments, actually only used information from 216 data points, in contrast, the results for sample size 216 in the non-bootstrapping experiments, are derived from 216×100 sample points. Only based on $1/100$ of information, bootstrapping produce similar results.

Conclusion

In this work, we devise a Bayesian structure learning method, using Markov Blankets to simplify and expedite the local structure recovery. The Markov blanket identification is realized by a fast feature selection based Grow-Shrink algorithm based on the assertion that Markov blanket is

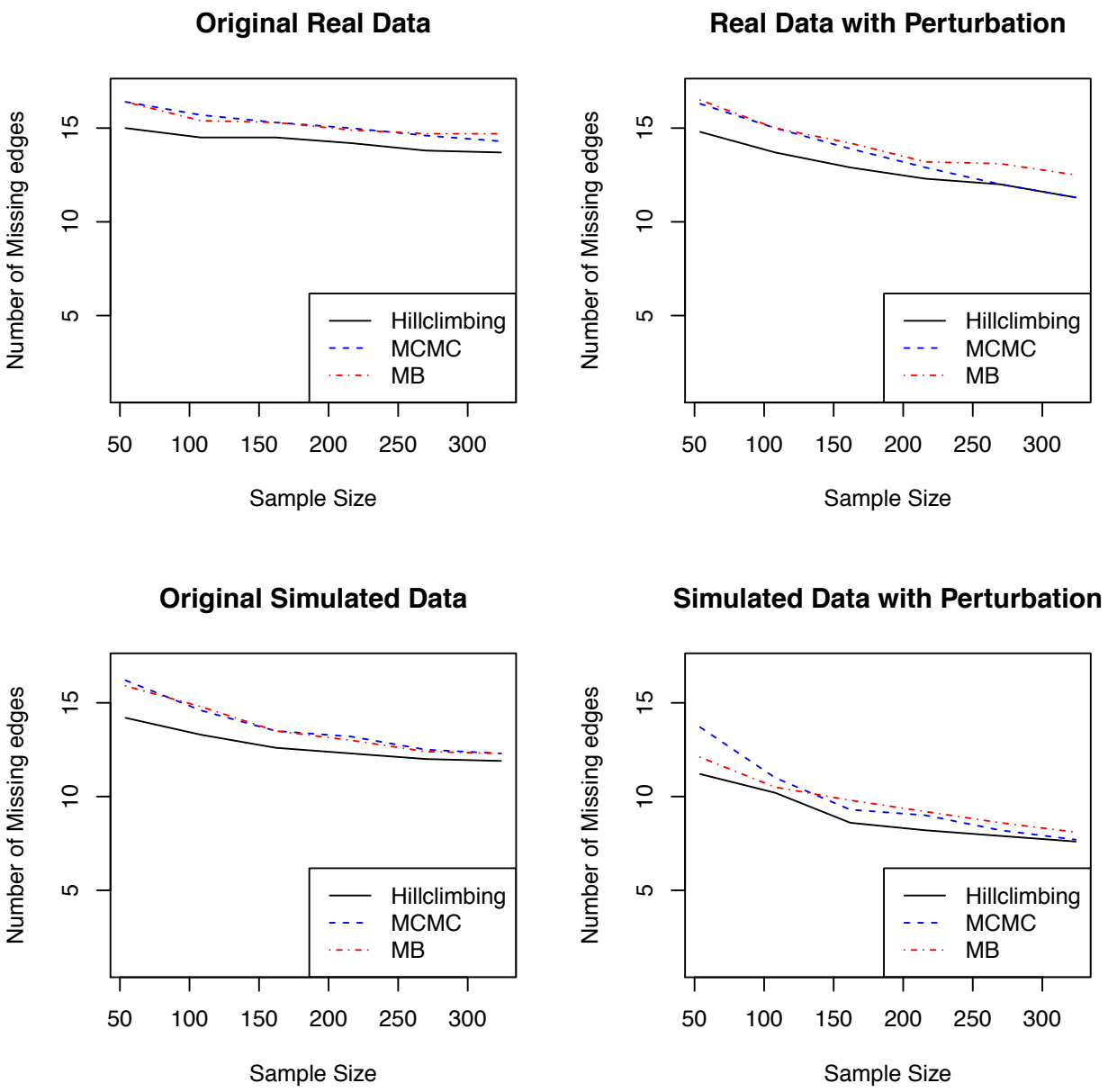


Figure 4.3 – Plot of Sample Size vs number of Missing edges in a graphs learned by Hillclimbing, MCMC or our Markov Blanket algorithm

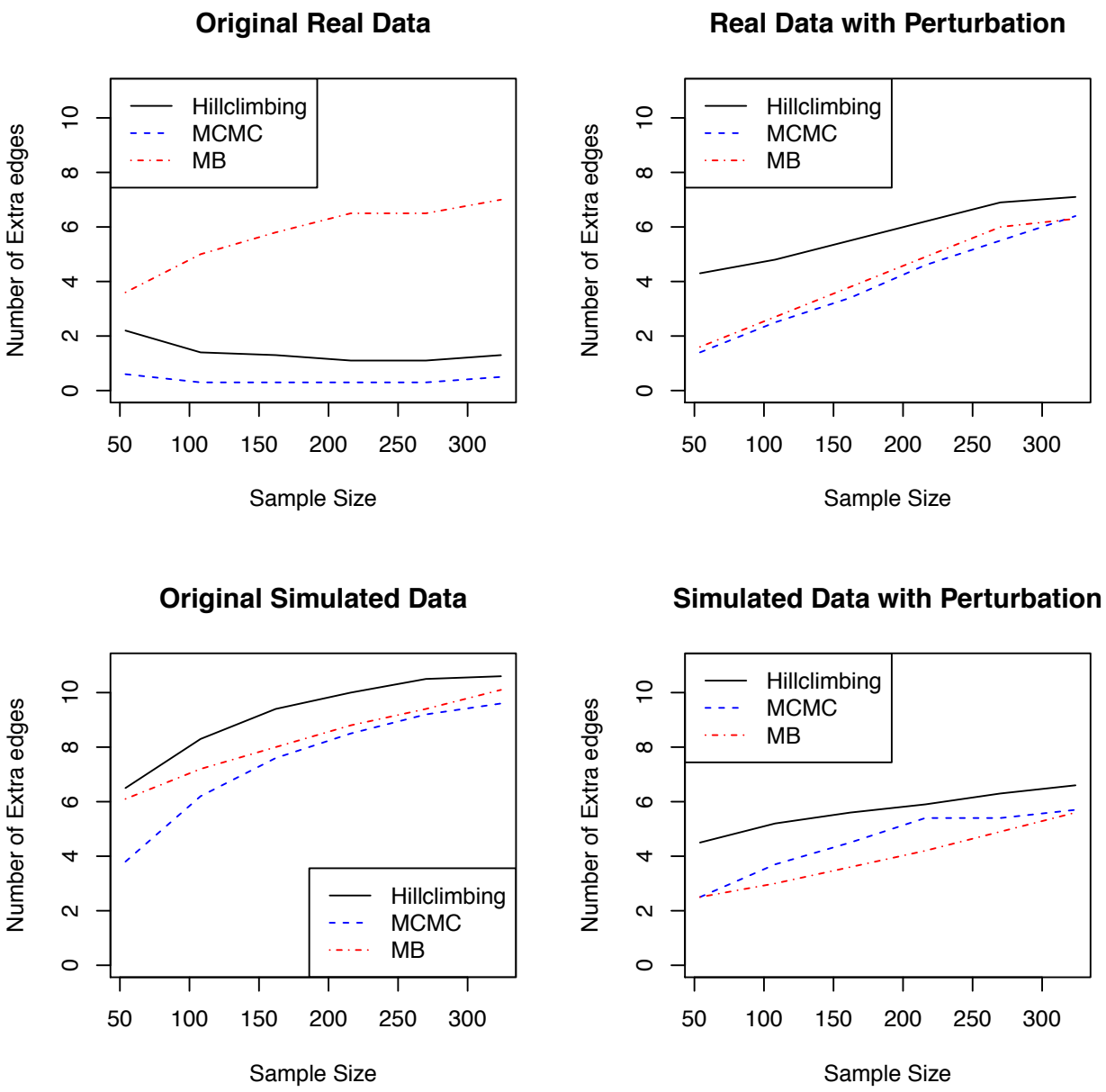


Figure 4.4 – Plot of Sample Size vs Number of Extra edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm

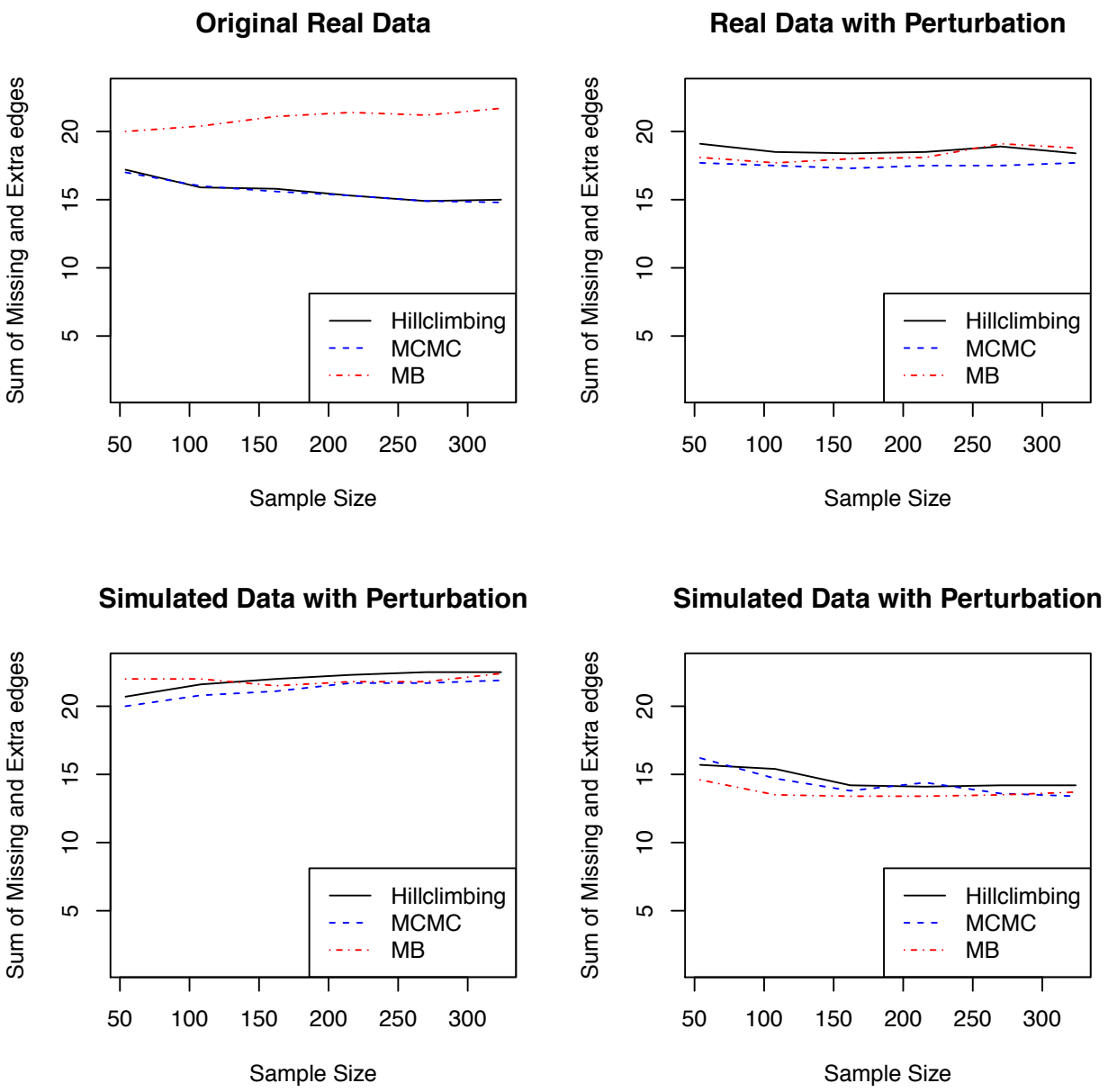


Figure 4.5 – Plot of Sample Size vs Sum of Missing and Extra edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm

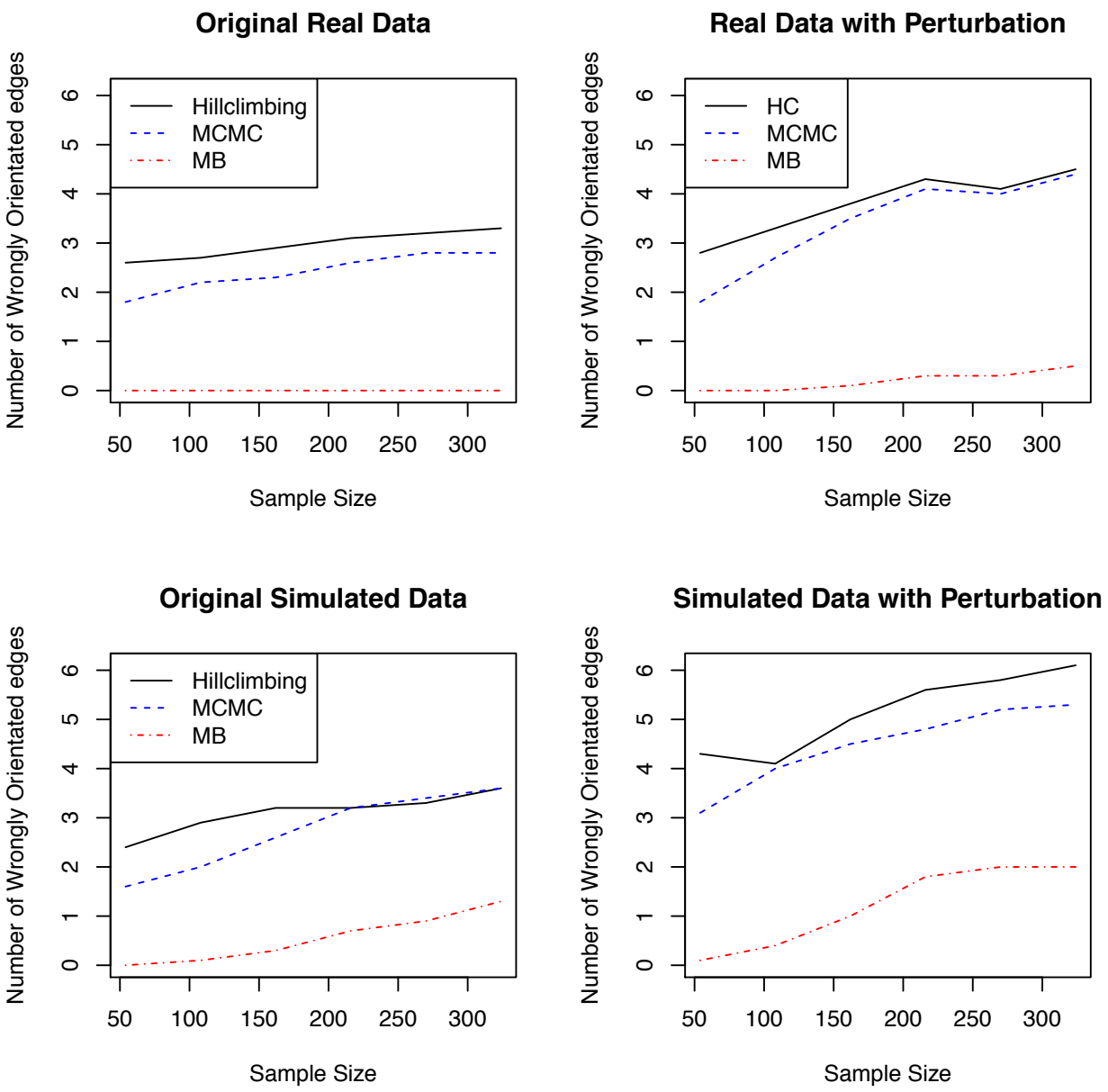


Figure 4.6 – Plot of Sample Size vs Number of Wrongly orientated edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm

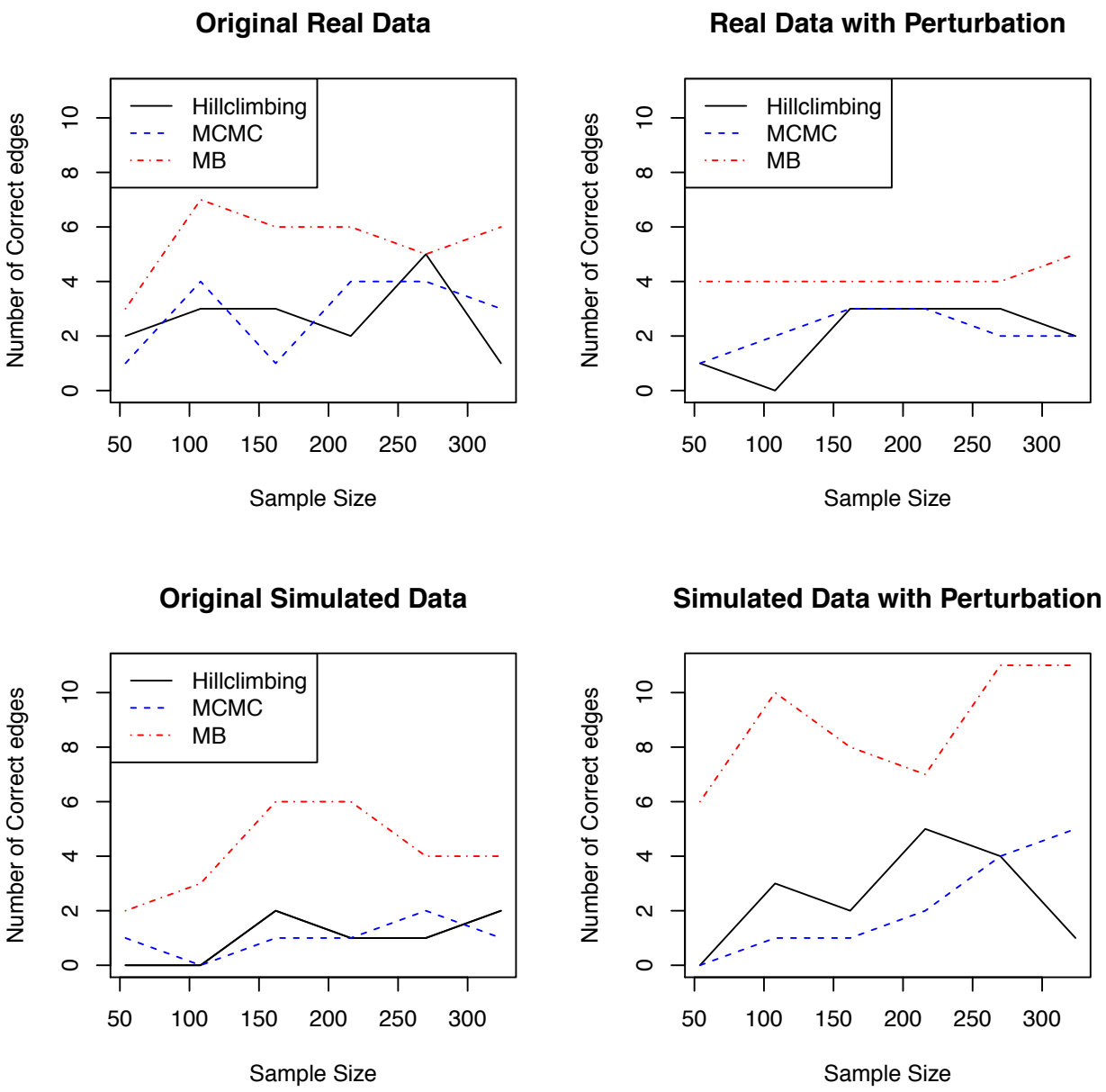


Figure 4.7 – Plot of Sample Size vs Number of Correct edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm, using Bootstrapping Resampling

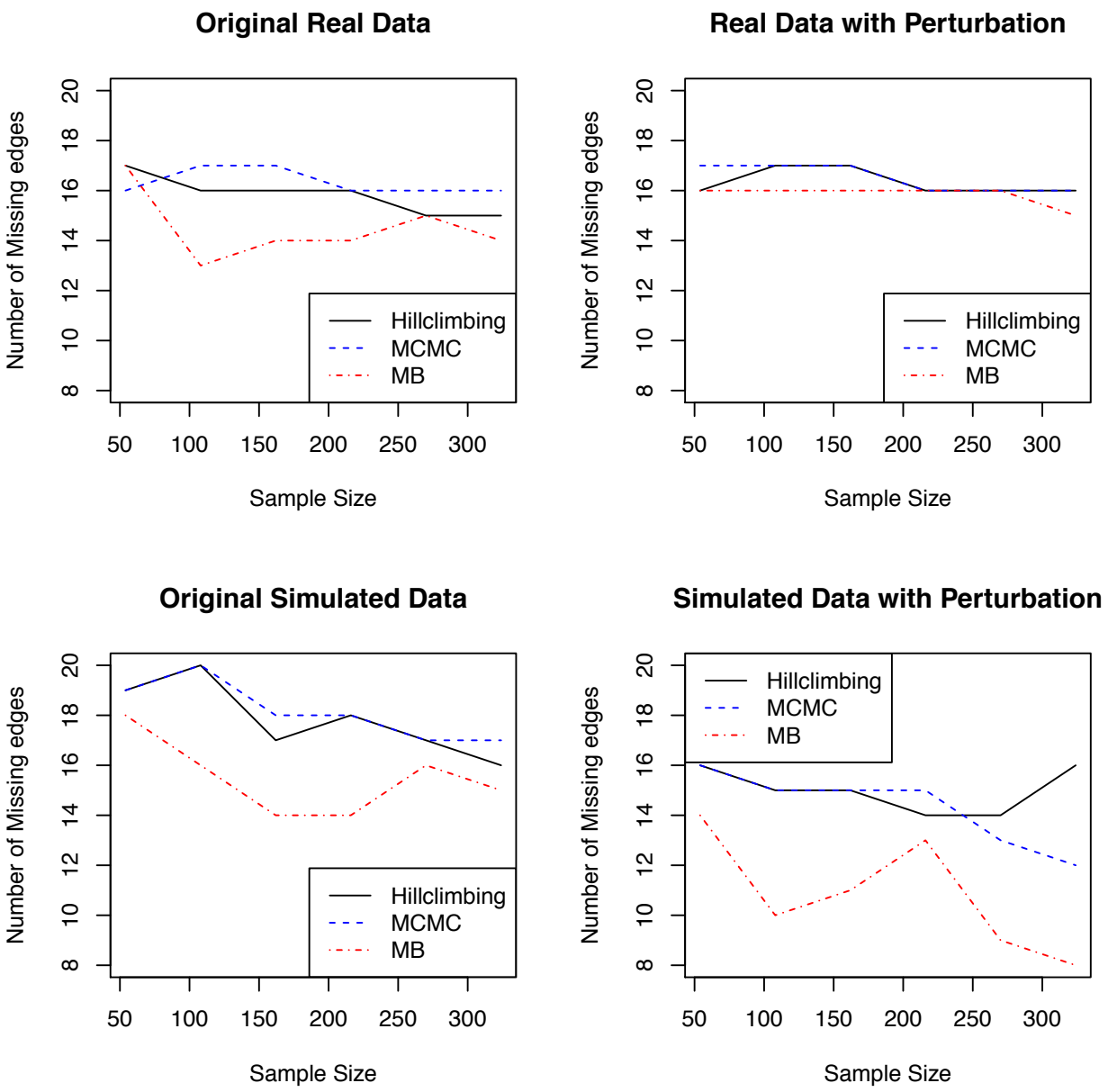


Figure 4.8 – Plot of Sample Size vs Number of Missing edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm, using Bootstrapping Resampling

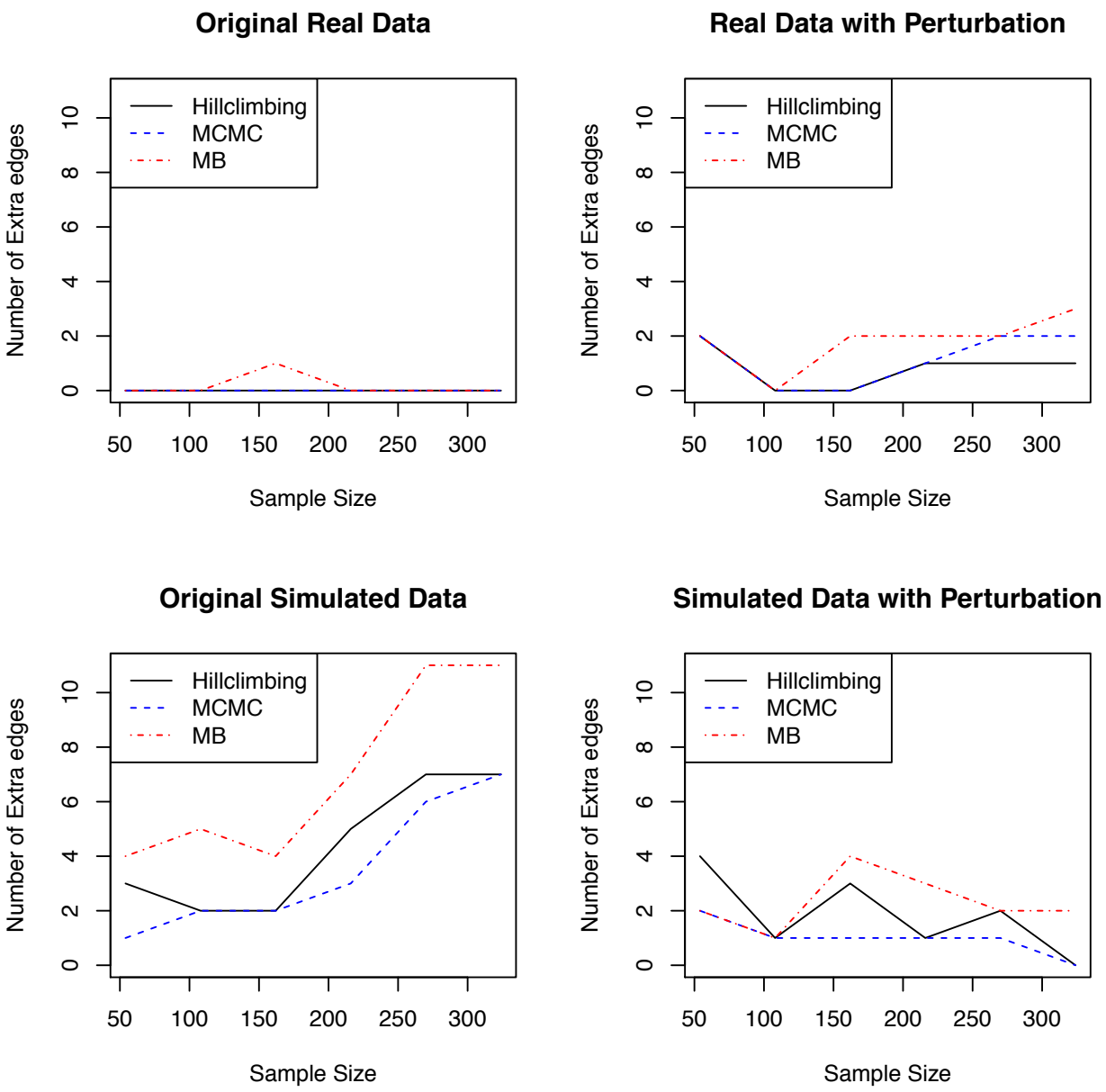


Figure 4.9 – Plot of Sample Size vs Number of Extra edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm, using Bootstrapping Resampling

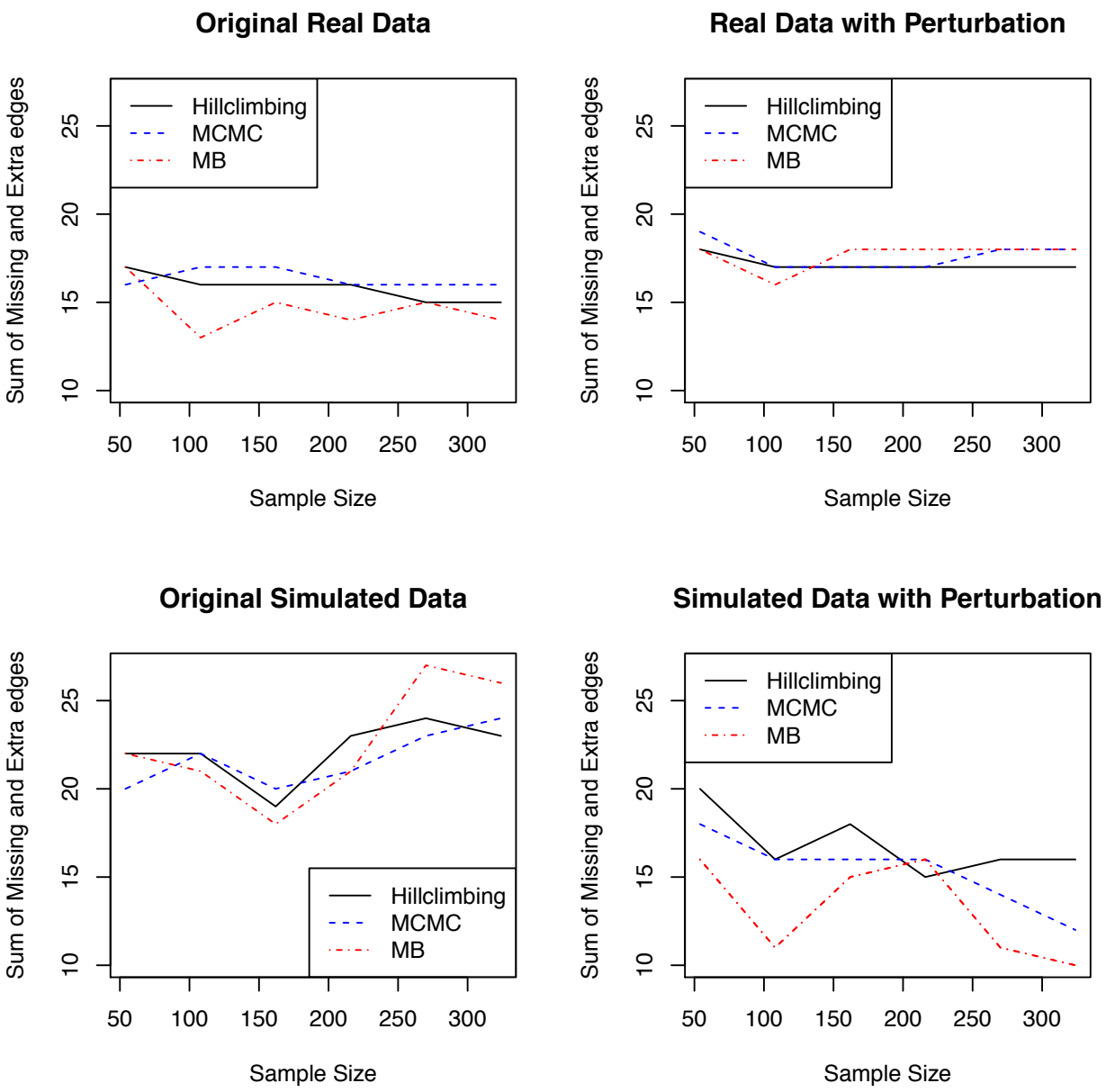


Figure 4.10 – Plot of Sample Size vs Sum of Missing and Extra edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm, using Bootstrap Resampling

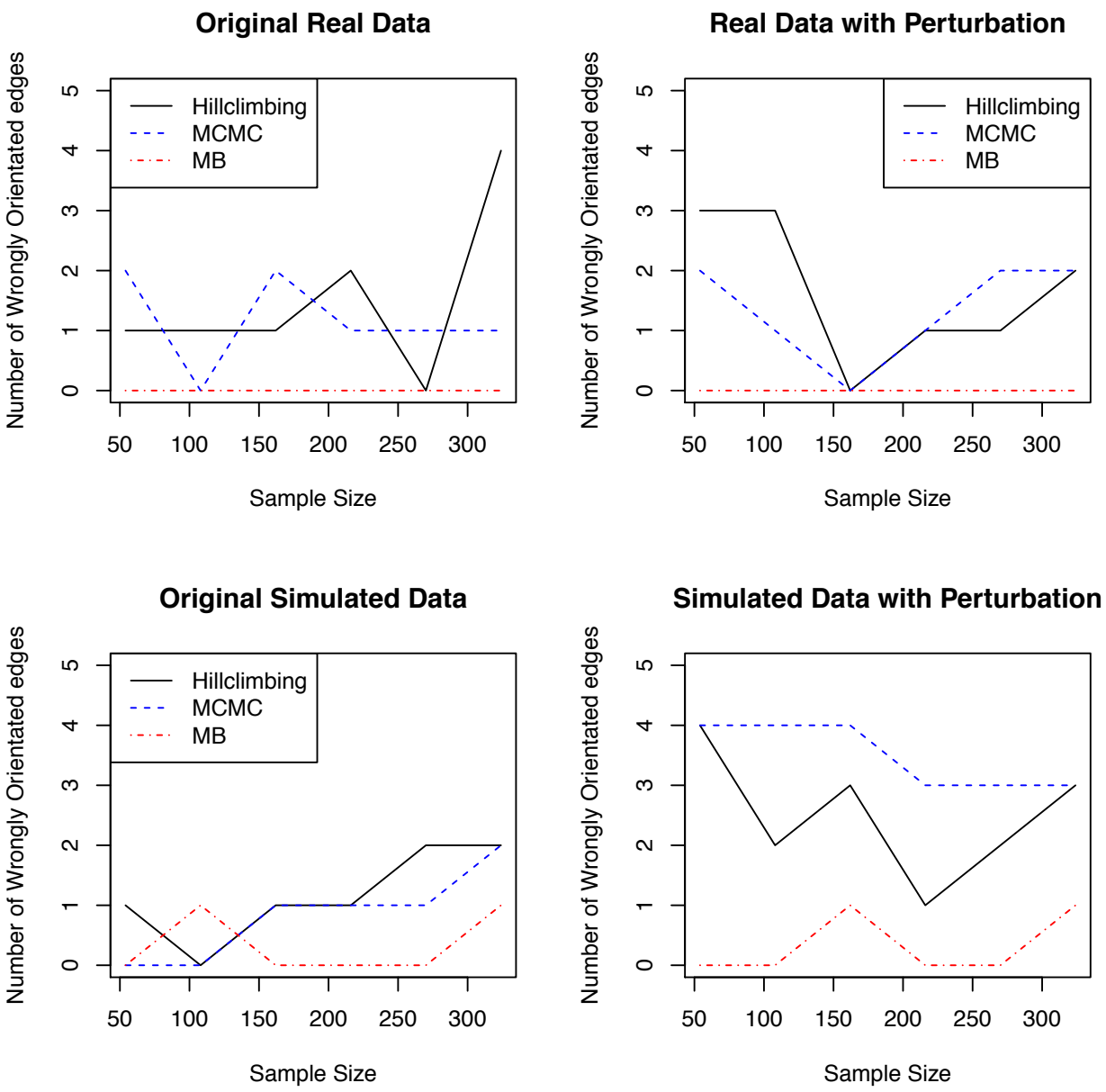


Figure 4.11 – Plot of Sample Size vs Number of Wrongly orientated edges in a graph learned by Hillclimbing, MCMC or our Markov Blanket algorithm, using Bootstrap Resampling

equivalent to strong feature set.

The algorithm is evaluated on a golden structure RAF pathway, compared with other popular BN learning algorithms. The empirical results show that our method produces competitive performance, even in limited sample size situation. The results also demonstrate that in our case, bootstrapping is successful when a sample size is very small.

5. K2-Based Markov Blanket Identification

5.1 Introduction

As we have studied in Chapter 4, the knowledge of Markov blankets of the nodes in a Bayesian network is valuable in recovering the local structures and then the whole network. The information of Markov blankets can also be applied in feature selection and classification.

Madden [89] argued that construction of a full Bayesian network for the purposes of classification may be computationally inefficient, as the whole structure may not be relevant to classification. Specifically, classification is unaffected by parts of the structure that lie outside the classification node's Markov blanket. MB as the strongly relevant nodes of a target, contains necessary and sufficient information to determine the behavior of the target. Thus, identifying the MB of the target in classification is desirable.

Koller and Sahami [90] first introduced Markov Blanket concept into the feature selection task. Intuitively, if a feature F_i is conditionally independent of the class label given some small subset of the other features, then we should be able to omit F_i without compromising the accuracy of class prediction. Koller and Sahami formalize this idea using the notion of a Markov blanket.

Definition 1 (Markov Blanket) [90]: For a feature set G and class label C, the set $M_i \subseteq G$ ($F_i \notin M_i$) is a Markov Blanket of $F_i (F_i \in G)$ if

$$F_i \perp G - M_i - F_i, C | M_i \tag{5.1}$$

This definition is equivalent to the one stated in Chapter 2.

In Chapter 4, we investigate the approach of feature selection, i.e., Grow and Shrink algorithm, to identify Markov blankets. Feature selection often involves correlation computation for dependence test, which is prone to errors. This disadvantage becomes even more severe when sample

sizes are limited, a common situation in biological data analysis.

In this work, we devise a new method for Markov blanket identification, using K2 algorithm - a theoretically sound, practically effective and efficient structure learning method. We investigate an asymptotic property of K2, which facilitates accurate construction of Markov blankets. Our method is compared with the approach proposed by Koller and Sahami (KS algorithm) [90] in Markov blanket identification on a golden structure network and in classification on a benchmark biological dataset.

The rest of the chapter is organized as follows: In Section 5.2, we introduce necessary background and review related works. Our method is discussed in details in Section 5.3. Section 5.4 shows the empirical comparison of our method and the KS algorithms. Finally, conclusions are drawn in Section 5.5.

5.2 Background and Related works

5.2.1 Related work

Madden [89] seeked to directly construct an approximate Markov blanket around the classification node. The algorithm involves three steps. In the first step, every node $X_i \in Z - X_c$ is tested relative to X_c to determine whether it should be considered to be a parent or a child of X_c as follows: identify the set containing the parents of the class variable and the set containing the children of the class variable, by comparing the overall probabilities of three networks: a new node added as the parent of the class variable, as the child and the current structure. The network with the highest probability will be kept as the base for the next node to be concerned. The second step is expanding the children set of the class variable by adding the direct parents of the direct children of the class variable, by running K2 algorithm. The third step is finding the dependence between the nodes in the children set, by constructing a restricted tree structure. After the structure is constructed, the conditional probabilities are calculated using K2 estimation. Then based on the posterior probabilities of each class given the other nodes in the structure, the predicted class is

assigned.

5.2.2 KS algorithm

Koller and Sahami [90] were the first to apply Markov Blanket concept into feature selection. They proposed a theoretical framework for optimal feature selection and formalized the problem as follows [91]:

Let G be a subset of the overall feature set F. Let f_G denote the projection of f onto the variables in G. Markov blanket filtering aims to minimize the discrepancy between the conditional distributions $P(C|F = f)$ and $P(C|G = f_G)$, as measured by a conditional entropy:

$$\Delta_G = \sum_f P(f) \cdot D(P(C|F = f)||P(C|G = f_G)) \tag{5.2}$$

where $D(P||Q) = \sum_x P(x) \cdot \ln(P(x)/Q(x))$ is the Kullback-Leibler divergence [92, 93], which is a non-symmetric measure of the difference between two probability distributions P and Q.The goal is to find a small feature set G for which Δ_G is small.

The following proposition due to Koller and Sahami [90] establishes the relevance of the Markov blanket concept to the measure ΔG .

Proposition: For a complete feature set F, let G be a subset of F,and $G' = G \backslash F_i$. If $\exists M_i \subseteq G$ (where M_i is a Markov blanket of F_i), then $\Delta G' = \Delta G$.

The proposition implies that once we find a Markov blanket of feature F_i in a feature set G, we can safely remove F_i from G without increasing the divergence to the desired distribution. Koller and Sahami [90] further prove that in a sequential filtering process in which unnecessary features are removed one by one, a feature tagged as unnecessary based on the existence of a Markov blanket M_i remains unnecessary in later stages when more features have been removed.

In most cases, however, few if any features will have a Markov blanket of limited size, and we must instead look for features that have an 'approximate Markov blanket.' For this purpose we

Table 5.1 – KS algorithm: identify a Markov Blanket

- ***Initialize:***
  - $G = F$
- ***Iterate*** until the size of G reached the pre-defined number (**m**):
  - For each feature $F_i \in G$, let M_i be the set of k features

    $F_j \in G - F_i$ for which the correlations between F_i and F_j are the highest
  - Compute $\Delta(F_i|M_i)$ for each F_i
  - Choose the F_i that minimizes $\Delta(F_i|M_i)$, and let $G = G - F_i$

define:

$$\begin{aligned}\Delta(F_i|M) &= \sum_{f_M, f_i} P(M = f_M, F_i = f_i) \times \\ &\quad D(P(C|M = f_M, F_i = f_i)||P(C|M = f_M))\end{aligned} \tag{5.3}$$

If M is a Markov blanket for F_i then $\Delta(F_i|M) = 0$. Since an exact zero is unlikely to occur, we relax the condition and seek a set M such that $\Delta(F_i|M)$ is small. Note that if M is really a Markov blanket of F_i, then we have $P(C|M,F_i) = P(C|M)$. This suggests an easy heuristic way to to search for a feature with an approximate Markov blanket.

Since the goal is to find a small non-redundant feature subset, and those features that form an approximate Markov blanket are most likely to be more strongly correlated to F_i, we construct a candidate Markov blanket for F_i by collecting the k features that have the highest correlations (defined by the Pearson correlations between the original nonquantized feature vectors) with F_i, where k is a small integer. We have the KS algorithm proposed by Koller and Sahami, [90]) depicted in Table 5.1

5.2.3 K2 algorithm

The well-known K2 framework for induction of Bayesian network from data was developed by Cooper and Herskovits [24]. We summarize their approach in two parts: (1) Determining network structure (2) Determining network probabilities.

5.2.3.1 Determining network structure

Cooper and Herskovits's framework is built on determining which of two Bayesian network structures is more likely. Assume D is a database of cases and Z is the set of variables represented by D. B_{S_i} and B_{S_j} are two belief-network structures containing exactly those variables that are in Z, then they aim to calculate $P(B_{S_i})/P(B_{S_j})$. According to Bayes' rule,

$$\frac{P(B_{S_i}|D)}{P(B_{S_j}|D)} = \frac{P(B_{S_i},D)/P(D)}{P(B_{S_i},D)/P(D)} = \frac{P(B_{S_i},D)}{P(B_{S_j},D)} \tag{5.4}$$

Therefore, the problem of calculating $P(B_S|D)$ reduces to computing $P(B_S,D)$. Cooper and Herskovits's equation for calculating $P(B_S,D)$ is based on four assumptions that they identified:

1. Variables are discrete and all are observed (i.e. there are no hidden or latent variables)
2. Cases occur independently, given a belief network model
3. There are no cases that have variables with missing values
4. The density function $f(B_P|B_S)$ is uniform; we are therefore indifferent regarding the prior probabilities to place on a network structure B_S

Let Z be a set of n discrete variables, where a variable X in Z has r possible value assignments. Let D be a database of m cases, where each case contains a value assignment for each variable in Z. Let B_S denote a network structure containing just the variables in Z. Each variable X_i in B_S has a set of parents, represented as a list of variables π_i. Suppose there are q_i such unique instantiations

of π_i. Let N_{ijk} be defined as the number of cases in D in which variable X_i is in k^{th} state and π_i is in j^{th} state. Let N_{ij} be defined as:

$$N_{ij} = \sum_{k=1}^{r_i} N_{ijk} \tag{5.5}$$

Then, given the assumptions outlined above,

$$P(B_S, D) = P(B_S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \tag{5.6}$$

Equation 5.6 can be combined with Equation 5.4 to give a computable method of comparing the probabilities of two network structures, when given a database of cases for the variables in the structures. Since, by the third assumption listed above, the prior probabilities of all valid network structures are equal, $P(B_S)$ is a constant. Therefore, to maximize $P(B_S, D)$ just requires finding the set of parents for each node that maximizes the second inner product of Equation 5.6. Cooper and Herskovits develop this into their *K2* algorithm which takes as its 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. Its output is a list of the parents of each node. The $K2$ algorithm works by initially assuming that a node has no parents, and then adding incrementally that parent whose addition most increases the probability of the resulting network. Parents are added greedily to a node until the addition of no one parent can increase the network structure probability. The function used in this procedure is taken from the second inner product of Equation 5.6:

5.2.3.2 Determining network probabilities

Cooper and Herskovits present a simple formula for calculating conditional probabilities, after the network structure has been found. Let θ_{ijk} denote the conditional probability that a variable X_i in B_S has the value v_{ik}, for some k from 1 to r_i, given that the parents of x_i, represented by π_i, are instantiated as w_{ij}. Then $\theta_{ijk} = P(X_i = v_{ik} | \pi_i = w_{ij})$ is termed a network conditional probability. Let ξ denote the four assumptions of Section 5.2.3.1. Then, given the database D, the Bayesian

network structure B_S and the assumptions ξ, the expected value of θ_{ijk} is given by:

$$E[\theta_{ijk}|D,B_S,\xi] = \frac{N_{ijk}+1}{N_{ij}+r_i} \tag{5.7}$$

$$g(i,\pi_i) = \sum_{j=1}^{q_i} \frac{(r_i-1)!}{(N_{ij}+r_i-1)!} \sum_{k=1}^{r_i} N_{ijk}! \tag{5.8}$$

In a single iteration of $K2$, an arc is added to node i from the node z that maximizes $g(i,\pi_i \cup z)$. If $g(i,\pi_i) > g(i,\pi_i \cup z)$, no arc is added.

5.2.4 Using a Bayesian Network for Classification

As pointed out by Friedman and Goldszmidt [94], inductive learning of general Bayesian networks is unsupervised in the sense that no distinction is made between the classification node and other nodes - the objective is to generate a network that 'best describes' the data. Of course, this does not preclude their use for classification tasks.

A Bayesian network may be used for classification as follows. Firstly, assume that the value of the classification node X_c is unknown and the values of all other nodes are known. Then, for every possible instantiation of X_c, calculate the joint probability of that instantiation of all variables in the network given the database D. Cooper and Herskovits' s formula for calculating the joint probability of a particular instantiation of all n variables is:

$$P(X_1 = x_i, \cdots, X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i | \Pi_i = \pi_i) \tag{5.9}$$

By normalizing the resulting set of joint probabilities of all possible instantiations of X_c, an estimate of the relative probability of each class instantiation is found. The vector of class probabilities may be multiplied by a misclassification cost matrix, if available. Then, the class instantiation with the highest probability is assigned as predicted class type.

5.3 Method

The $K2$ algorithm [24] is a greedy search algorithm that learns the network structure of Bayesian Networks (BN) from the data. It attempts to select the network structure that maximizes the network's posterior probability, given the experimental data. The *K2* algorithm reduces computational complexity by requiring a prior ordering of nodes as input, from which the network structure will be constructed.

The ordering required in the *K2* algorithm is such that if node X_i comes prior to node X_j in the ordering, then node X_j cannot be a parent of node X_i. In other words, the potential parent set of node X_i can include only those nodes that precede it in the input ordering. In *K2*, the candidate parents Pa_i for node X_i is initially set to the empty set. The algorithm visits each node X_i according to the sequence specified in the prior node ordering and greedily adds Pa_i to the parent set of node X_i if the addition of the parent to the node X_i maximizes the score of the network. The algorithm stops when any of the following conditions are met:

- The maximum number of parents for that particular node has been reached.
  (This number is specified for each node. A good number for this is $n-1$).

- There are no more legal parents to add.

- No parent addition improves the score.

In this work, we use *K2* algorithm with random ordering in a novel way to discover the direct links in Bayesian Networks.

With very large sample size, K2 always finds an edge (undirected) regardless of ordering if the edge truly exists in a BN. This is formally described in the following theorem.

Theorem 1: Assume the number of cases in the database $\mathscr{D}$ generated from a Bayesian belief network $\mathscr{G}$ with positive conditional probabilities, approaches infinity. An edge between two nodes can be identified by *K2* algorithm in all node-orderings if and only if these two nodes are adjacent in $\mathscr{G}$.

Figure 5.1 – Illustration for the theorem of K2 extension

We now provide an intuitive explanation of the theorem's correctness (refer to the appendix - page 88, for a complete proof of the theorem).

On one hand, we show if an edge exists in a BN, for instance, the edge between X_1 and X_2 in Figure 5.1, $K2$ can detect it in any ordering. For a given ordering, either X_1 is in front of X_2 or reverse, $K2$ can find either $X_1 \in Pa_2$ or $X_2 \in Pa_1$, i.e., detect $e(X_1, X_2)$ in all cases. On other hand, we show that if two nodes don't have direct link (e.g., X_2 and X_4), in some ordering, $K2$ doesn't return the link between the two nodes. In our case, X_2 and X_4 are related through X_3. In the orderings where X_3 is after both X_2 and X_4, $K2$ can find $e(X_2, X_4)$; but in the ordering where X_3 is before X_2 as in the true BN, X_2 is blocked away from X_4 by X_3, then $K2$ can't see $e(X_2, X_4)$. Combining these two aspects, we argue that $K2$ can always detect the edges if and only if the edges truly exist in a BN.

5.3.1 Identifying Markov Blanket

Parents and Children

Theorem 1 provides an approach to detect true links between two nodes, namely, the connections between a node and its parents and children. We run $K2$ multiple times (2000 in our experiments) in random ordering. If an edge between the target X and a node Y exists in all learnt structures from any ordering, it is a true edge and Y is either a parent or child of X.

Spouse

So far the only missing nodes in MB are spouses of the target node. Spouses are related through their child (see Figure 5.2). Y and X are spouses because of their common child Z. We denote the set of parents and children of the target Y, $PC(Y)$, which has been found by $K2$ learning. If a node $X \in V - PC(Y)$, a node $Z \in PC(Y)$, $X \perp Y$ and $X \not\perp Y|Z$, then X is a spouse of Y through Z. In

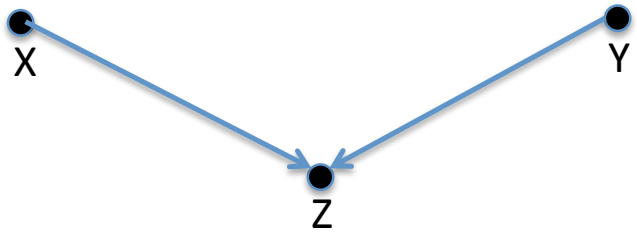


Figure 5.2 – V structure shows spouses and child relationship

the experiments, we apply partial correction and Fisher's Z-transform for independence test.

5.4 Experimental Results

In this selection, we present empirical comparison of our method and *KS* algorithm on both accuracy of identifying Markov Blanket and classification.

5.4.1 Identify Markov Blankets

Data

The ALARM network is a medical diagnostic system for patient monitoring. It consists of 37 nodes and 46 edges connecting them. The random variables in the ALARM network are discrete in nature. The number of discrete states depends on the node. The random variables in the network can take two, three, or four states (Figure 5.3).

Based on the given probabilities for the network, we generated 11 datasets with different sample sizes (from 10,000 to 60). Node 32 is chosen as our target node since it has the most complicate neighborhood, *Mb(32) ={31 33 34 35 36 10 14 4}* , in which the first 5 are direct neighbors (parent or child) and the rest 3 nodes are spouses.

In *KS* algorithm, we need to specify two parameters - k (MB size of individual nodes) and size of feature set (the number of features to keep). We set $k = 2$ since it produces the best results in this experiment, and size Of feature set to be 8, which is the actual size of $Mb(32)$. For our method, the only parameter concerned is confidence level α for independence test when identifying spouses. In this study, we set $\alpha = 0.05$.

Results

The Markov blankets of Node 32 identified on each dataset are listed in Table 5.2. To evaluate induction results, we use sensitivity and specificity. Sensitivity is defined as the ratio of matching positives between predicted and observed over all observed positives. Specificity is defined as the ratio of matching negatives between predicted and observed over all observed negatives.

In this case, positives are the nodes in MB, while negatives are the nodes not in MB. For Node 32, numbers of real positives and negatives are 8 and 28 respectively. The learnt Markov blankets from all datasets and all methods are listed in Table 5.2.

As we can see in Table 5.2 and Figure 5.4, when sample size is sufficient, both KS and our method correctly identify all 8 nodes in MB with both sensitivity and specificity to be 1. When sample size drops to 800, KS only detect 5 true positives, while our method still produces perfect MB. When sample size is 60, sensitivity drops to 50% for KS, and 63% for our method. One interesting observation is that our method produce only one (Node 37 which is secondary neighbor of the target) or none false positive with all datasets. When sample size is very small (80, 60), the sensitivity is still perfect for our method. Overall, for all 11 datasets in both sensitivity and specificity categories, our method outperforms KS methods (Figure 5.4).

5.4.2 Classification

Data

We use molecular database [95] from UCI Repository to evaluate our method's performance for classification. 106 instances in the dataset have 57 attributes, which are 57 sequential nucleotide ('base-pair') positions, each with 4 values of state (A, G, T, C). Each instance is labeled as promoter or not promoter.

KS algorithm

We randomly partition the whole dataset into three subsets at an approximate ratio of (3 : 1 : 1), used as training set (60 instances), validation set (20 instances), and testing set (26 instances), respectively. The three datasets have roughly the same ratio of positive (promoter) to negative (not

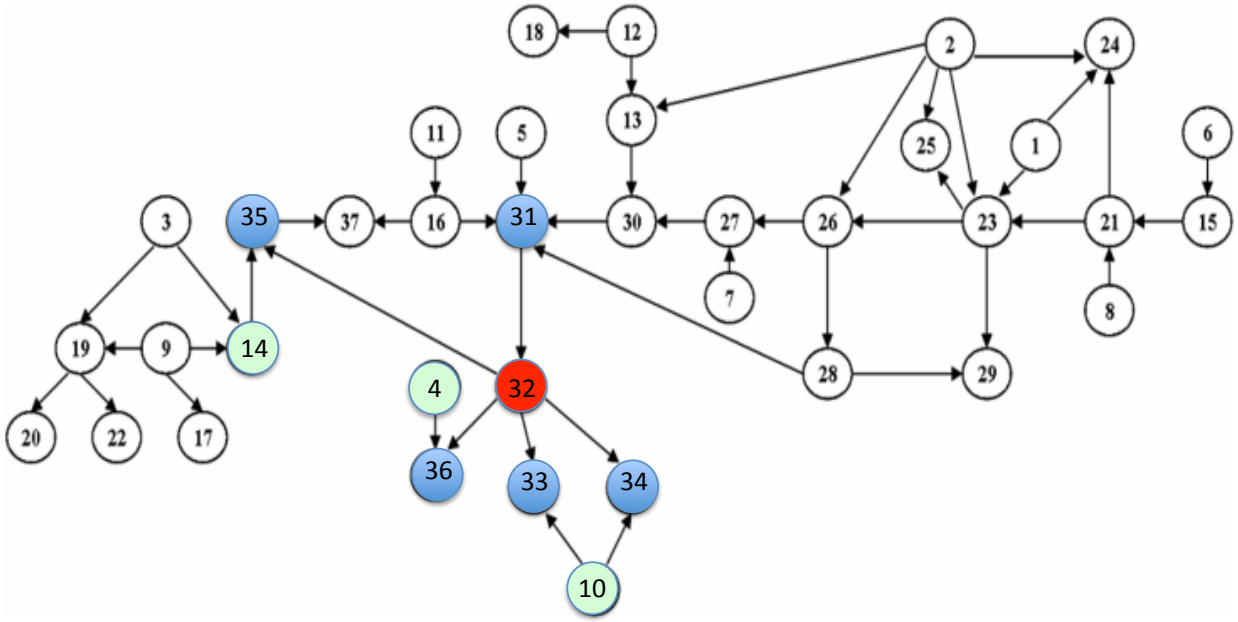


Figure 5.3 – Illustration of the Markov Blanket of Node 32 in the Alarm network: Mb(32)={ 36, 35, 34, 33, 14, 10, 4}, the first 5 nodes in blue are parents and children of Node 32, the other 3 nodes in green are spouses.

Table 5.2 – Markov Blankets of Node 32 in the ALARM network learned by KS algorithm and our extended K2 algorithm from a wide range of sample sizes from 60 to 10,000

Sample Size	**Markov Blanket of Node32**	
	KS	K2 Extended
10000	**36 35 34 33 31 14 10 04**	**36 35 34 33 31 14 10 04**
5000	**36 35 34 33 31 14 10 04**	**36 35 34 33 31 14 10 04**
800	**36 34 31 10 04** 29 26 24	**36 35 34 33 31 14 10 04**
500	**36 35 33 31 14 10 04** 22	**36 35 34 33 31 14 10 04** 37
400	**36 34 33 31 10** 29 25 20	**36 35 34 33 31 14 10** 37
300	**36 35 33 14 10 04** 29 22	**36 35 31 14 04** 37
200	**36 35 14 10** 30 29 25 20	**36 35 34 33 31 14 10** 37
150	**36 35 34 14 10 04** 29 24	**36 35 34 33 31 14 10**
100	**36 35 33 14 10** 20 18	**36 35 34 33 31 14** 37
80	**36 35 34 14 04** 22 20 18	**36 35 34 33 31**
60	**36 35 34 14** 29 22 20 16	**36 34 33 31 10**

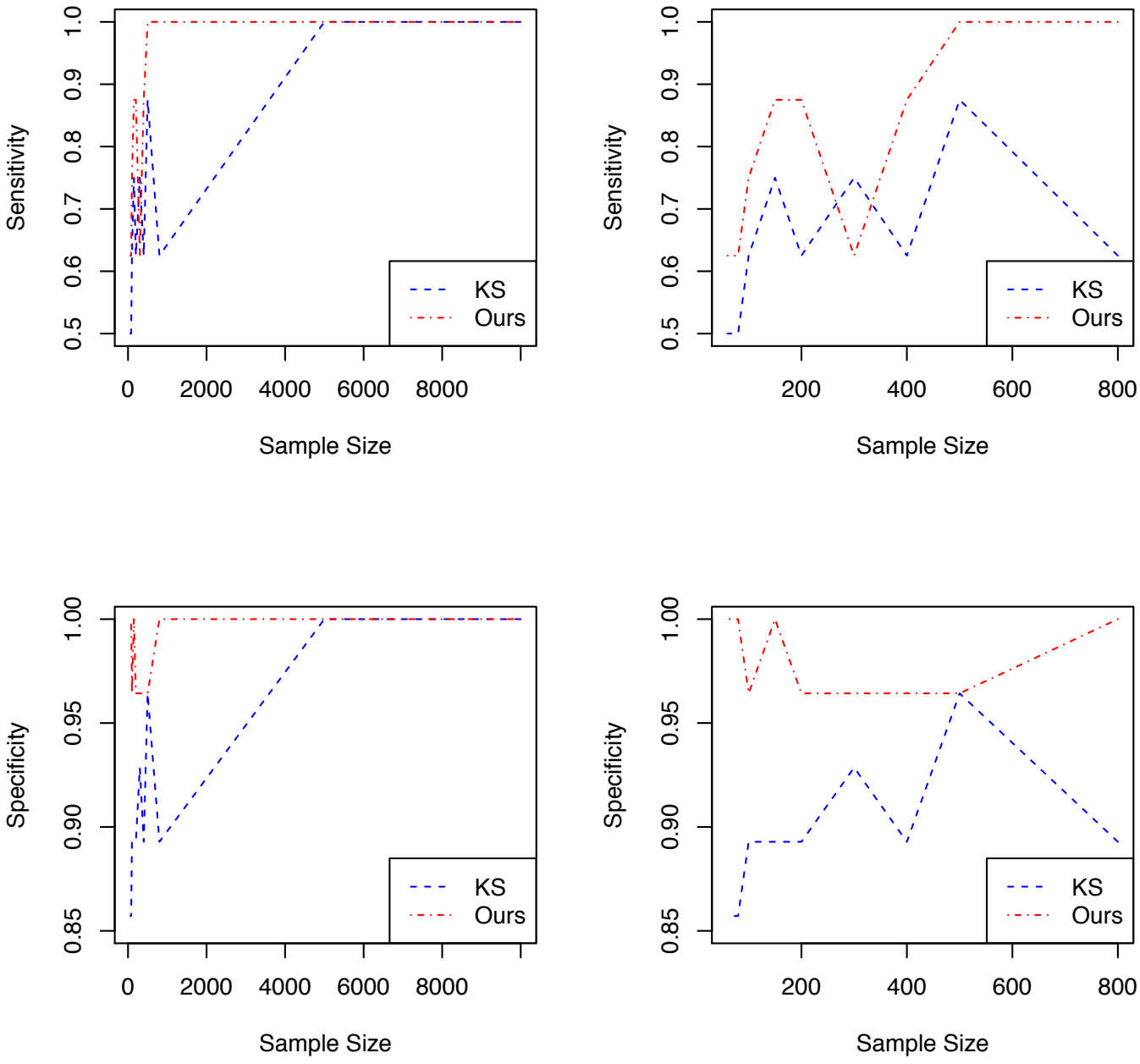


Figure 5.4 – Accuracy Comparison of two Markov Blanket learning algorithms (KS and ours) in terms of Sensitivity and Specificity, testing on Node 32 in the Alarm network. The nodes in the true Mb(32) are considered as positives, all the others are treated as negatives.

Table 5.3 – The results of feature selection and classification using KS and our methods on the Molecular dataset

	KS	**Ours**
MB(C) positions	{19, 40}	{ 7, 16, 17, 18, 19, 40 }
Correctly classified	18 (69%)	23 (88%)
Incorrectly classified	8 (31%)	3 (12%)

promoter). The validation set is set aside for selecting the optimum size of feature set in the KS algorithm.

Markov blanket of the class variable ($Mb(C)$) is learnt using KS algorithm for all candidate feature sizes (1 to 10, in this study). Each resulting Markov blanket ($Mb(C)$) is then applied as the feature set for Bayesian classifier training. Learned classifiers are evaluated on the validation set. The feature size which produces the best classification result on the validation set is chosen as the feature size parameter (m) in KS algorithm.

The validation set is added to form new expanded training set (80 instances) to identify Markov blanket of the class variable ($Mb(C)$) using KS algorithm with the feature size m determined in the previous step. The Bayesian classifier is trained with learnt $Mb(C)$ as the feature set , and evaluated on the testing set.

Our Method

In this study, we set the only parameter $\alpha = 0.05$, the confidence level of independence test using partial correlation and Fisher-Z transform. The expanded training set formed in the above KS experiment are used to train our model, then tested on the same testing set as the KS experiment.

Results

The feature size which produces the best performance in the KS algorithm is $m = 2$. As Table 5.3 is shown, when sample size is small in our study (80 training instances), our method performs better than KS algorithm (8% higher accuracy than the best result from KS).

5.5 Conclusions

In this chapter, we developed a K2-based approach to detect links between variables. Our method can quickly and accurately identify Markov blankets in the graph. Empirical results demonstrate that our method obtains more accurate Markov blankets than the representative MB learning algorithm (KS algorithm). When the learned MB is applied as the feature set for classification, our method also outperforms KS algorithms on the benchmark Molecular dataset. Furthermore, when sample sizes are small, our method obtains better performance than KS algorithm for both tasks of MB identification and feature selection.

6. Tree-augmented naïve Bayesian for protein-protein interaction assessment

6.1 Introduction

Proteins are fundamental to many biological processes from DNA replication and transport to signal transduction [96]. Investigations of protein functions have been the growing efforts in diverse disciplines, such as biochemistry, molecular medicine, structural biology, and computational biology. In general, proteins do not act isolated, instead through protein-protein interactions (PPI) in living cells. Identification of protein-protein interactions is crucial to functional genomics. The known function of a protein can disclose a clue on defining the functions of other proteins within the same interacting protein complex. PPI networks also find important applications, for instances, mapping cellular pathways and their intricate cross-connectivity, and discovering new drugs [97–99].

Computational approaches developed over the years for protein-protein interaction prediction differ in feature information and methodologies. Estimation of interaction sites, analysis of genomic sequences, and protein domain information are some of the frequent subjects for PPI prediction in the earlier stage. To evaluate interaction sites, Kini and Evan [100] investigated in recognizing specific residue motifs, Jones and Thornton [101] studied solvent accessible surface area and hydrophobicity, and Bock and Gough [102] explored protein primary structure and associated physicochemical properties. For genomic sequence analysis, researchers examined correlated mutations in amino acid sequences between interacting proteins [103], conservation of gene neighborhoods and gene order [104], gene fusion method or 'Rosetta stone' [105, 106], genomic context to infer functional protein interactions [107], and the principle of co-evaluation [108–110]. Protein domain information is considered for PPI prediction with correlated sequence-signatures as mark-

ers [111], statistical prediction [112] and integrative approach [113], and random decision forest framework [114].

Recently, integrating complementary data for PPI prediction has been demonstrated prospective, and has gained an increasing interest. Integrative methods including probabilistic decision tree [115], logistic regression model [116], Bayesian network [117], and naïve Bayesian classifier [118, 119], have been employed to a variety of complementary data in different scales. Zhang et al. [115] predicted co-complexed protein pairs from high-throughput protein interaction datasets and twelve major categories of gene- and protein-pair characteristics. Zhong and Sternberg [116] combined five features of identical anatomical expression, phenotype, function annotation, microarray co-expression, and the presence of orthologs. Jansen et al. [118] incorporate direct experimental PPI data, mRNA expression, gene function, and protein essentiality in a naïve Bayes model. Myers et al. [117] integrated information of interaction data from PPI databases, cellular localization data, transcription factor binding sites from the Saccharomyces cerevisiae Promoter database, and biological complex curated literature from the Saccharomyces cerevisiae Genome Database to discover query-based pathway-specific networks. Rhodes et al. [119] employed a naïve Bayesian network to combine ortholog interaction datasets, gene expression data, shared biological function, and enriched domain pairs.

A naïve Bayesian classifier with the strong assumption of conditional independences among all the attributes given the value of the class variable, is surprisingly competitive with state-of-the-art classifiers, such as C4.5 in many cases [120–123]. Friedman [123] explained that both the bias and variance components of the estimation error impact classification performance, and under certain conditions, the low variance associated with the naïve Bayesian classifier can dramatically reduce the effect of the high bias that results from the strong independence assumptions. A naïve Bayesian classifier also performs better than unrestricted Bayesian classifiers in some situations. Friedman et al. [120] argued that Bayesian scoring functions were designed for structure learning, not particularly for classification tasks. The scoring functions measure the errors of the learned Bayesian network over all the variables in the domain, hence, minimizing this error does not necessarily

minimize the local error in predicting the class variable given the attributes, especially when there are many attributes. The naïve Bayesian classifier considers the contributions from all other attributes to the class variable because all of the other attributes as the class variable's children are in its Markov blanket.

In order to further improve the performance of the naïve Bayesian classifier without sacrificing its characteristics of simplicity and robustness, Friedman et al [120] first proposed a tree-augmented naïve Bayesian (TAN) classifier, which is based on the structure of naïve Bayes, requiring that the class variable be a parent of every attribute, and also allows additional edges between attributes that capture correlations among them, where the augmenting edges are in a reasonable restricted tree-like form for the seek of computational cost.

In this paper, we extend our previous work [124] on a Bayesian network-based integrative method for PPI prediction. We introduce a classification - oriented Bayesian network, namely, tree augmented naïve Bayes (TAN) into PPI prediction. In our experiments, the features used for predicting PPI in a target organism (e.g. human), are extracted from model organisms (e.g., Saccharomyces cerevisiae, C. elegans, and Drosophila melanogaster). The advantage of a cross-organism predictive model is that model organisms are well studied and have a nearly unfathomable amount of experimental data, while there may be little information about the target organism, especially about newly sequenced proteins. Our TAN based model integrates multiple microarray gene expression datasets and gene ontology (GO) (The Gene Ontology Consortium, 2000) information from three model organisms, and explicitly incorporates in the confidence scores of ortholog mappings between the target organism and the model organisms.

The rest of the paper is organized as follows. First, we describe TAN algorithm, feature extraction and a Manually constructed Bayesian network classifier in Methods. Then, we present experiments on human PPI prediction in Results. Finally, we state some concerns and make concluding remarks in Discussions.

6.2 Methods

6.2.1 TAN

In a tree-augmented naïve Bayes (TAN) , the class variable as the root of the tree has no parents and each attribute has the class variable and at most one other attribute as its parents, therefore, each attribute can have one augmenting edge (edge connecting two attributes) pointing to it. To learn the TAN model, we follow the algorithm proposed by Friedman et al. [120] and based on a well-known method reported by Chow and Liu [125] for learning tree-like Bayesian networks (also refer to Pearl's book [83]). By taking advantage of the restriction in a TAN model, the algorithm reduces the problem of constructing a maximum likelihood tree to finding a maximal weighted spanning tree in a graph. A maximal weighted spanning tree can be constructed by selecting a subset of arcs from a graph such that the selected arcs constitute a tree and the sum of weights attached to the selected arcs is maximized, which can be solved in the standard procedures with the time complexity of $O(n^2 \log n)$, where n is the number of vertices in the graph [126]. The TAN model construction algorithm shown in Table 6.1 has time complexity of $O(n^2 N)$, where N is the number of instances in data D.

6.2.2 Feature Selection

6.2.2.1 Orthologous mapping score: *S*

For each protein-protein pair (P_1, P_2) in the target organism, we first identify its orthologous pairs ($R_1^{(i)}$, $R_2^{(i}$)) in three model organisms ($i = 1,2,3$), using Inparanoid algorithm introduced by Remm et al. [127] [32]. Ortholog clusters are seeded with a two-way best pairwise match, and then grown by gathering inparalogs independently around the seed. Each member of the constructed protein cluster receives an inparalog score between 0 and 1.0, which reflects the confidence about the orthologous relationship, also called orthologous mapping score ($S^{(i)}, i = 1,2,3$) in this paper. We discretize $S^{(i)}$ into three states: low, medium and high.

Table 6.1 – TAN Classifier Construction Algorithm

1. Compute the conditional mutual information between each pair of variables, given the class variable. We set A_1 to A_n as n attributes, C as the class variable

$$I_p(A_i,A_j|C) = \sum_{i\neq j,c} P(a_i,a_j,c) \times \log \frac{P(a_i,a_j|c)}{P(a_i|c)P(a_j|c)}$$

2. Build a complete undirected graph in which the vertices are the attributes A_1,.., A_n. Annotate the weight of an edge connecting A_i to A_j by $I_p(A_i,A_j|C)$.
3. Build a maximum weighted spanning tree.
4. Transform the resulting undirected tree to a directed one by choosing a root variable and setting the direction of all edges to be outward from it.
5. Construct a TAN model by adding a vertex labeled by C and adding an arc from C to each A_i.

6.2.2.2 Microarray feature: *M*

Microarray gene expression data disclose functional associations among proteins [118, 119, 128]. The correlation between expression profiles of two proteins is measured with Pearson correlation coefficient (*PPC*) [129], which is defined as:

$$r = \frac{1}{n-1} \sum_{i=1}^{n} (\frac{X_i - \bar{X}}{s_X}) \cdot (\frac{Y_i - \bar{Y}}{s_Y}) \tag{6.1}$$

where $\frac{X_i - \bar{X}}{s_X}$, $\bar{X}$ and s_X are the standard score, sample mean and sample standard deviation.

For each orthologous pair ($R_1^{(i)}$, $R_2^{(i}$)), three *PPC*s are calculated from three microarray datasets of the i^{th} organism. The absolute values of the raw *PPC*s are then discretized according to a 4-level (LL, L, H, HH) uniform quantization scheme. The three resulted discrete *PPC*s of the pair are combined into one value ($M^{(i)}$), which has 20 possible states, because we consider the contributions from the three individual microarray datasets are somehow related since their origins are the same

organism.

6.2.2.3 GO features: *F*, *P* and *C*

First, three orthologous GO features are developed from each of the three domains: 'molecular function', 'biological process', and 'cellular component'. Those features indicate the association between a pair of proteins in different aspects. The first feature is the total number of shared terms between the two proteins. The second feature is the correlation ratio, which is defined based on the number of gene products in common. The third feature is based on minimum GO distance between two proteins in GO structures. Apparently, the three features are related to each other, which leads us to join the three features into one variable with 32 states. Consequently, the orthologous pair $(R_1^{(i)}, R_2^{(i}))$ in the i^{th} organism of a target protein pair (P_1, P_2) has been generated three GO features, namely, $F^{(i)}$, $P^{(i)}$ and $C^{(i)}$ derived from 'molecular function', 'biological process', and 'cellular component' respectively (refer to [124] for details).

6.2.3 A manually constructed Bayesian network (*MC*)

In our previous work [124], we empirically demonstrate that a manually constructed Bayesian network classifier (*MC*) provides better classification accuracy than a naïve Bayes classifier (*NB*). To evaluate the TAN classifier, we compare it with the *MC* model and the naïve classifier. Here, we give a brief review of the *MC* model; and the details can be found in [124]. The *MC* model shown in Figure 6.1 integrates microarray features $M^{(i)}$, GO features ($F^{(i)}$, $P^{(i)}$, $C^{(i)}$) and orthologous mapping confidence ($S^{(i)}$) into a Bayesian network, where protein-protein interaction is modeled as the two- state class variable, denoted as *I*. In order to distinguish the three model organisms, instead using superscripts in the feature names, we prefix feature name with organism initial, i.e., *F* for fruit fly (D. melanogaster), *W* for worm (C. elegans) and *Y* for yeast (S. cerevisiae), for instance, *FS* stands for mapping score for fruit fly. As we can see, orthologous mapping confidences are explicitly modeled into the classifier; when given mapping confidence and *PPI* (or *I*), microarray and GO features within the same organism are conditional independent; when given *PPI*,

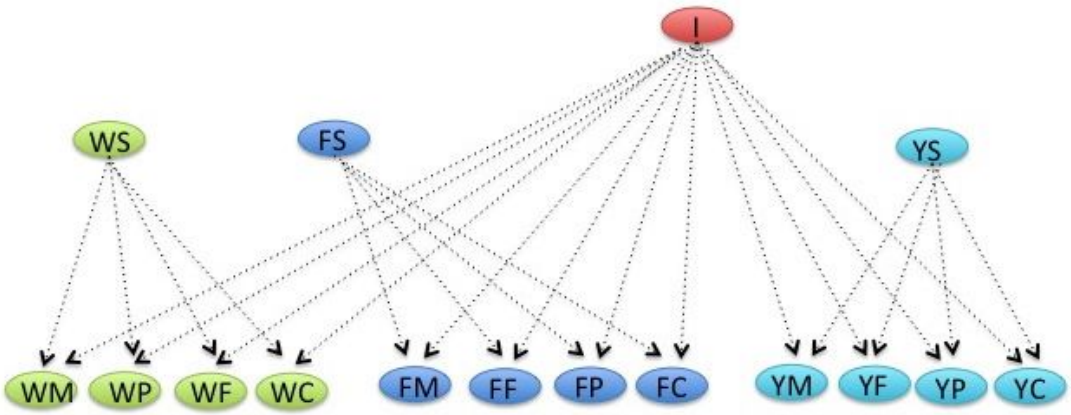


Figure 6.1 – A manually constructed Bayesian network classifier

features from different organisms are conditional independent. This cross-organism conditional independence allows us to derive a simple solution for PPI prediction, as we detail next.

The Bayesian approach to classify a test sample is to assign the most probable class, or the class with a larger posterior probability for a two-class problem. For the model shown in Figure 6.1, we have the ratio of the posterior probability for two classes:

$$L = \frac{P(I=1)\cdot\sum_{i=1}^{3} P(S^i)\cdot P(M^i,F^i,P^i,C^i|S^i,I=1)}{P(I=0)\cdot\sum_{i=1}^{3} P(S^i)\cdot P(M^i,F^i,P^i,C^i|S^i,I=0)} \tag{6.2}$$

where, (based on conditional independence shown in Figure 6.1)

$$P(S^i)\cdot P(M^i,F^i,P^i,C^i|S^i,I) = P(M^i|S^i)\cdot P(P^i|S^i,I)\cdot P(C^i|S^i,I)\cdot P(F^i|S^i,I) \tag{6.3}$$

For a pair of proteins, we compute its probability ratio L and predict the two proteins as an interacting pair if $L > 1$ and non-interacting pair otherwise. The prior $P(I=1)$ and $P(I=0)$ can be computed empirically.

6.2.4 ROC Analysis

Although accuracy estimation is very widely used in the Machine Learning community for comparison of classifiers, Provost et al. [130] have argued that accuracy estimation is not the most

appropriate metric when cost and class distributions are not specified precisely. As an alternative, they propose the technique of Receiver Operating Characteristic (ROC) analysis, which is taken from signal detection theory. In the Machine Learning context, a ROC graph is a plot of false positive rate against true positive rate. A deterministic classifier produces a single point in ROC space, but a probabilistic classifier such as those considered in this work produces a curve. As stated by Provost and Fawcett [131], the benefit of ROC curves is that they illustrate the behavior of a classifier without regard to class distribution or error cost.

6.3 Results

6.3.1 Data

The 10,163 known human interacting pairs also with mapped orthologs in any model organism are collected from the HPRD [132, 133] database. Since non-interacting protein data are not available, the negative samples are randomly generated. A protein pair is considered to be a negative sample if the pair does not exist in the interaction set. Total of 209,761 human protein pairs are obtained as negative samples. The ratio of negatives and positives is about $20:1$. About $2/3$ of positive and negative data are separated into training data and the remaining samples are used as testing data. The final training set has 6,766 positives and 139,864 negatives and testing set contains 3,397 positives and 69,897 negatives.

Genome-wide orthologous mappings between human and the three model organisms (S. cerevisiae, C. elegans, and D. melanogaster) are downloaded from the Inparanoid database [127]. For each protein pair in human, we form a list of ortholog pairs in the model organisms along with mapping scores.

Microarray gene expression datasets are collected from NCBI Gene Expression Omnibus (GEO) [134]. As shown in Table 6.2, three microarray data sets for each model organism are obtained [135–141].

Table 6.2 – Microarray datasets used in the experiments, *N* represents the number of samples in each dataset

Organism	Dataset(*N*)	Dataset(*N*)	Dataset(*N*)
Yeast	GDS1115(131)	GDS465(90)	GDS92(40)
Worm	GDS1319(123)	GDS770(20)	GDS6(29)
Fruit fly	GDS2272(36)	GDS516(26)	GDS2673(27)

6.3.2 Human PPI prediction

We use *ROC* (receiver operating characteristic or plot of 1-specificity $\sim$ sensitivity) curves to evaluate the performance of a classifier. A *ROC* is obtained by varying a decision threshold, which is equivalent to adjusting the priors in a Bayesian network based classifier. We define specificity as the percentage of matched non-interactions between the predicted set and the observed set over the total number of observed non-interactions, and sensitivity as the percentage of matched interactions over the total number of observed interactions.

6.3.2.1 Classifier comparison

The TAN model, the manually constructed Bayesian network (*MC*) and the naïve Bayesian (*NB*) classifier are employed to predict human PPI. We implement *MC* and *NB* in Java, and use WEKA (http://www.cs.waikato.ac.nz/ml/weka/) package for TAN. The models are learned on the training set containing instances with at least one orthologous mapping information, described in Section 6.3.1. Then the obtained models are evaluated on the testing dataset also with one or more orthologous mappings and at the same ratio of positives and negatives. In case of missing values, the contribution from the attributes is ignored.

The results are presented in *ROC* curves in Figure 6.2. As we observe, TAN outperforms both the naïve Bayesian method and the manually constructed BN. With specificities fixed at approximately 70%, TAN, manual constructed and the naïve Bayesian method can achieve sensitivities at around 76%, 73% and 66% respectively.

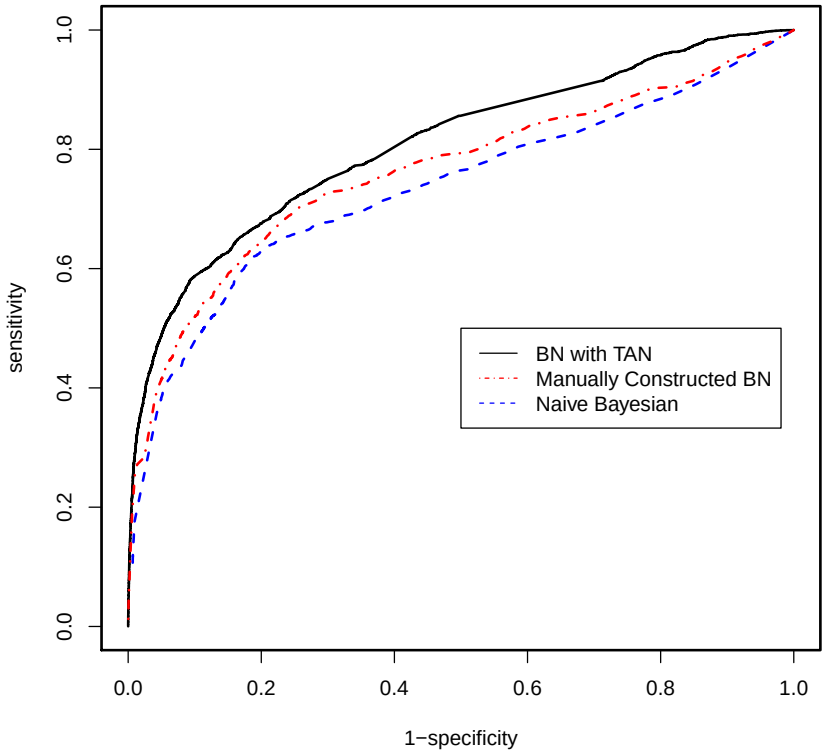


Figure 6.2 – A manually constructed Bayesian network classifier

Table 6.3 – Data Statistics

Mapping #	Training				Testing			
	Pos.	**Neg.**	**Total**	**Ratio**	**Pos.**	**Neg.**	**Total**	**Ratio**
$\geq$ **1**	7,164	142,973	150,137	**1**	3,581	71,486	75,067	**1**
1	3,473	104,302	107,775	**.72**	1,714	52,085	53,799	**.72**
2	2,943	32,557	35,500	**.24**	1,469	16,426	17,895	.24
3	748	6,114	6,862	**.05**	398	2,975	3,373	**.05**

6.3.2.2 Impact from missing data

Examining the data closely, we find a high proportion of missing values. As shown in Table 6.3, 72% of instances have only one ortholog organism information, 24% have two orthologous mapping, and only 5% have all ortholog mappings. The ratios are the same in training and testing datasets.

In this experiment, we train TAN models by instances with one orthologous mapping, two mappings and three mappings separately, and obtain Model1, Model2 and Model3 respectively. We then evaluate the models on the testing instances with same number of mappings as the models' training instances, e.g., Model3 is learned with 3-mapping instances, we evaluate it on 3-mapping testing instances. The results are displayed as *ROC* curves in Figure 6.3. As expected, the system's performance is getting better as more orthologous evidences are available. When the information is available from all three organisms, with the specificity of 70%, it can achieve 88% sensitivity; when two or one orthologous organisms is available, with the same specificity, its sensitivity can only be 72% or 66% respectively.

6.3.2.3 Study on the TAN structure

The results in Section 6.3.2.1 demonstrate that the TAN classifier achieves better classification performance than the manually constructed Bayesian network classifier, which is crafted based on our knowledge of the domain. In this experiment, we explore the connections, which the MC network has missed but the TAN model may reveal from data.

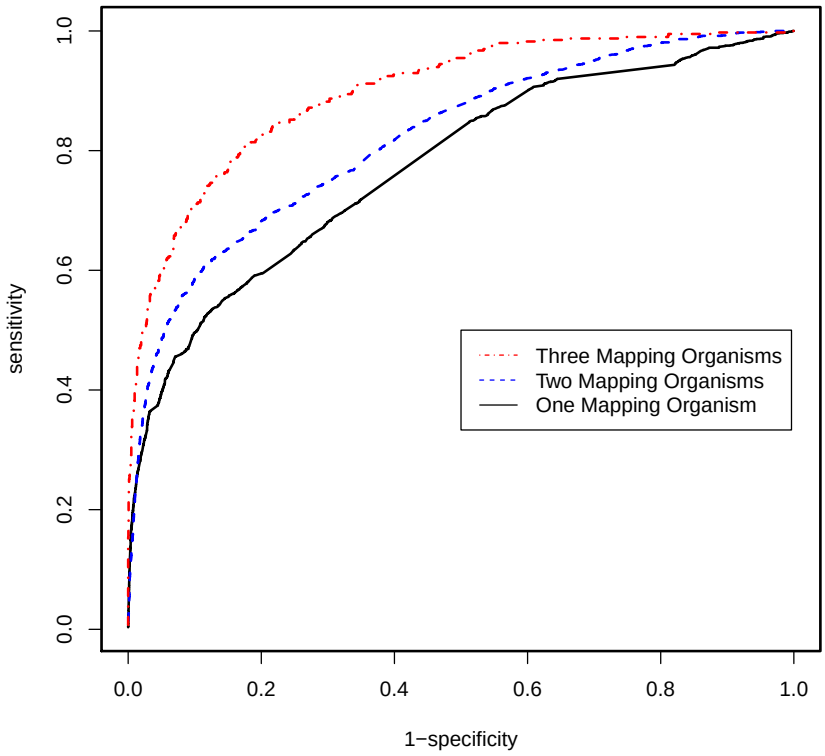


Figure 6.3 – ROC curves evaluating TAN models trained and tested on samples with N orthologous organisms, where N is 1, 2, or 3

Figure 6.4 shows the TAN structure learned from the origin imbalanced training data with at least one orthologous organisms. As TAN algorithm defined, every attribute has the class variable I and at most one other attribute as its parents, i.e., correlations among attributes are allowed. In Figure 6.4, we observe most of the augmented edges (12 out 14) are within the same organisms. Interestingly, the two cross-organism links (red edges) are within the same categories: $WS \rightarrow YS$ and $WF \rightarrow FF$. We wonder if the characteristics of imbalance (positive: negative = 1 : 20) and high missing rate (70% with one orthologous organism) in training data cover any interesting correlations.

We only keep the samples with all three orthologous organisms in the training data, and obtain 748 positives and 6,114 negatives. To balance the number of positives, we partition the negatives into 8 groups, and combine each group with the positives to form 8 training datasets. Each training dataset are used separately to learn TAN model. We observe all learned models outperform the model learned using all negatives, which is superior over the model trained with at least one mapping (refer to Section 6.3.2.2). We observe two trends in the eight learned TAN structures: (1) mapping scores are always on top of other attributes, i.e., the directions of the augmented edges are from the mapping scores to the others; (2) near half of the augmented edges are with the same categories, e.g., $YF \rightarrow FF$, $WC \rightarrow FC$. Figure 6.5 shows one of the structures. As we see, if there is a link between score and other attributes, its direction is always from score to the other; 6 out of 14 augmented edges are within the same categories (red links).

6.3.2.4 Study on the importance of attributes

The results from the last section suggest that orthologous mapping scores play special roles in the TAN model. In this experiment, we remove the entire mapping score attributes, and build the TAN model by using the rest of the 16 attributes from the balanced training dataset without missing orthologous info (described in Section 6.3.2.3). The resulting TAN structure is actually a naïve Bayesian network as shown in Figure 6.6. This suggests that the association among other attributes should go through the mapping scores, which agrees with the *MC* model (refer to Figure 6.1).

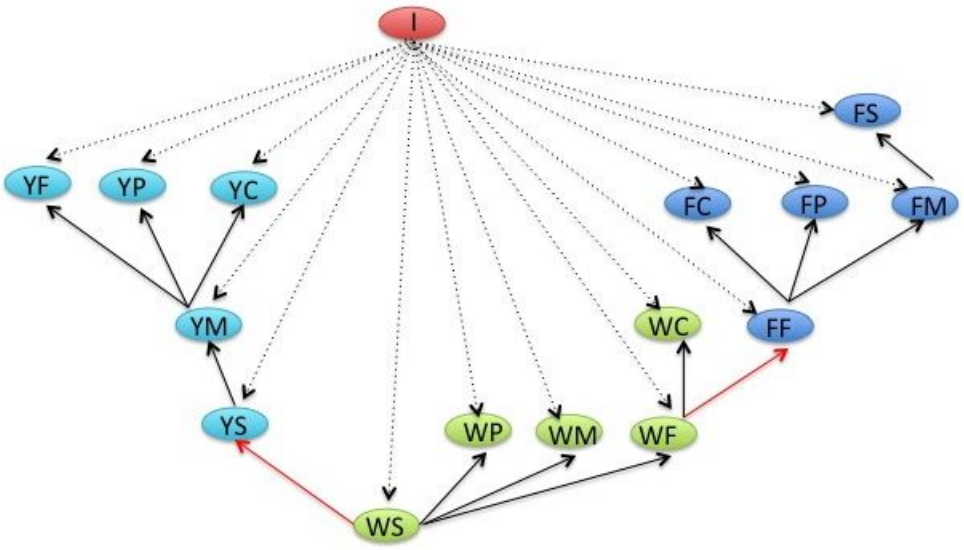


Figure 6.4 – The TAN structure learned from at least one ortholog organism and unbalanced training data

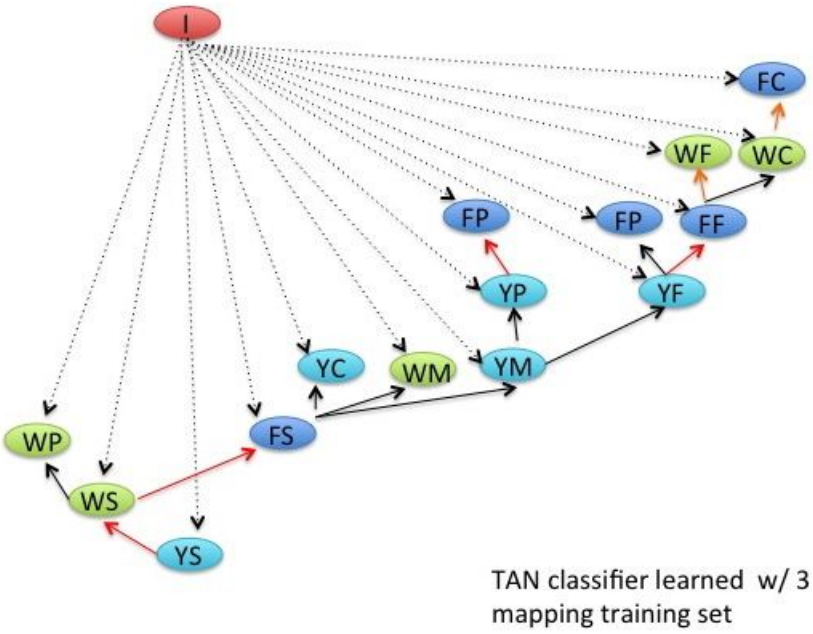


Figure 6.5 – The TAN structure learned from three ortholog organisms and balanced training data

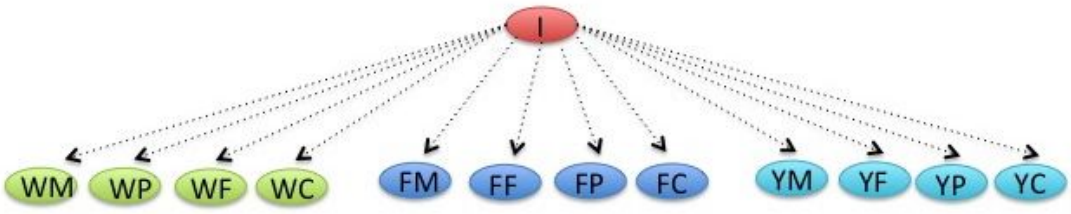


Figure 6.6 – The TAN structure learned from three ortholog organisms and balanced training data without the mapping score attributes

6.4 Discussions

6.4.1 TAN and unrestricted Bayesian network classifier

TAN relaxes the strong assumption of a naïve Bayesian classifier, but still has a restricted tree-like structure for the sake of computational efficiency. An easy question to ask is if we want to trade the computational cost for a better classification accuracy, can we achieve the aim by learning an unrestricted Bayesian network classifier?

Our empirical results reveal that an unrestricted Bayesian network classifier learned by hill-crimbing is even worse than a naïve Bayesian classifier. The *ROC* areas of unrestricted Bayesian, naïve Bayesian and TAN are 0.759, 0.780 and 0.808 respectively.

The scoring functions for structure learning cause the observed results [120], since they evaluate how closely each candidate network represents the probability distribution embedded in the training data, by measuring the error of the learned Bayesian network over all the variables in the domain, instead of the relevant features (the *Markov Blanket*) of the class variable. Thus, minimizing this error does not necessarily minimize the local error in predicting the class variable given the attributes.

6.4.2 TAN and imbalanced data

In many cases, imbalanced class distribution causes poor performances from standard classification algorithms [142, 143] which induce classifiers that maximize the overall classification accu-

racy. Well-known strategies to deal with the imbalance-class problem include resizing training data sets, adjusting misclassification costs, and recognition-based learning (learning from the minority class). By contrast, some evidences [143] support that rebalancing the classes artificially does not have a great effect on the predictive performance. Our data have heavy skewed class distribution (positives: negatives = 1 : 20). We studied how much rebalanced training data improve the performance of the induced TAN classifier.

We split the negative training sample into 20 equal-sized groups, and then combine positives in training with each of the negative groups to form 20 training datasets (7164 positives and 7149 negatives in each set). We then apply TAN on each of the training sets. The results show that training on balanced data can obtain slightly better classifiers (*ROC* area: 0.818 ± 0.002) than the TAN classifier (ROC area: 0.808) learned from imbalanced data. We didn't observe apparently improving nor worsening in the naïve Bayesian classifiers learned from rebalancing data. The result suggests that the TAN classifier is slightly more sensitive to rebalancing than the naïve Bayesian classifier.

6.4.3 TAN and missing values

Table 6.3 tells us half of the positives in the training data have only one orthologous mapping. Missing values can strongly impair the learned classifier. In WEKA implementation for TAN, attributes with missing values are ignored when computes the likelihood of each class value. To better handle missing values problem, Friedman [123] proposed a variant of *EM* for selecting the graph structure that can efficiently search over many candidates for the case of TAN models. Studying the feasibility of this and other methods to better handle missing values is one of my interests for future work.

6.4.4 Small sample size

To learning the parameters of a network we estimate conditional probabilities in the form of $\hat{P}_D(X|\Pi_X)$, which can be realized by partitioning the training data according to the possible values

of Π_X and then computing the frequency of X in each partition. When some of these partitions contain very few instances, however, the estimate of the conditional probability is unreliable. This problem is not as acute in the case of a naïve Bayesian classifier, since it partitions the data according to the class variable, and usually all values of the class variables are adequately represented in the training data. In TAN networks, however, for each attribute we assess the conditional probability given the class variable and another attribute. This means that the number of partitions is at least twice as large. Thus, it is not surprising to encounter unreliable estimates, especially in small data sets.

The standard practice in Bayesian statistics in this situation is a *smoothing operation* on the parameters learned. In Bayesian learning of a multinomial distribution, with the assumptions of *Dirichlet* priors and using marginal to estimate conditional probabilities, we obtain the following formula to calculate the conditional probability given the class variable and another attribute:

$$\theta^s(x|\Pi_x) = \frac{N \cdot \hat{P}_D(\Pi_x)}{N \cdot \hat{P}_D(\Pi_x) + N^0_{x|\Pi_x}} \cdot \hat{P}_D(\Pi_x) + \frac{N^0_{x|\Pi_x}}{N \cdot \hat{P}_D(\Pi_x) + N^0_{x|\Pi_x}} \cdot \theta^0(x|\Pi_x) \tag{6.4}$$

where, $\theta^0(x|\Pi_x)$ is the prior estimate of $P(x|\Pi_x)$ and $N^0_{x|\Pi_x}$ is the confidence associated with that prior. Note that this application of *Dirichlet* priors biases the estimation of the parameters depending on the number of instances in the data with particular values of X's parents. Thus, it mainly affects the estimation in those parts of the conditional probability table that are rarely seen in the training data. Investigation of smoothing operation in small sample applications using Bayesian network is one of my future interests.

6.4.5 Concluding remarks

Knowledge of protein-protein interactions is crucial to understand biological processes. Enormous biological data generated by advanced technologies are available. It is desirable to develop effective and efficient computational approaches to integrate heterogeneous data sources and predicting large-scale PPIs. In this paper, we extended our previous work of using a Bayesian classifier to

predict human PPI from data containing microarray expression measurements and GO features of orthologous protein-protein pairs in model organisms.

To further improve prediction accuracy, we introduce tree-augmented naïve Bayes (TAN) classifier. A TAN classifier relaxed the strong assumption of conditional independence among all variables given the state of the class variable, however it maintains the simplicity and robustness of the naïve classifier.

The empirical evaluations show that the TAN performs much better in term of classification accuracy, even better than our previous manual created Bayesian classifier. The more orthologous information from testing samples available, the more accurate the prediction is. When testing instances have orthologous values from all three organisms, with the reasonable specificity of 70%, sensitivity can achieve 88%. The TAN structure conforms some important links as shown in the manual construction, e.g., microarray and GO features are descendants of the mapping scores, it also reveals some correlations among attributes, which are embedded in data.

7. Conclusion

Recently, Bayesian network formalism as a graphic representation of the dependence relationships among a group of variables in an uncertainty domain, has seen growing applications in Bioinformatics. The sound theoretical foundation and fruitful researches in practice have made Bayesian networks a valuable tool to handle biological data analysis. A Bayesian network provide a innate way to present biological networks with the characteristics of uncertainty and dynamic; it naturally integrates different types of resources and existing knowledge in aim to discover new causal relationships or inferencing desired unknowns. However, with the development of new technologies, massive high-throughput data challenge the researchers to design new effective and efficient algorithms for Bayesian network induction and inference.

7.1 Summary and Contribution

In this thesis, I develop new methods for BN structure learning with applications to biological network reconstruction and assessment. The first application is to reconstruct the genetic regulatory network (GRN), where each gene is modeled as a node and an edge indicates a regulatory relationship between two genes. In this task, we are given time-series microarray gene expression measurements for tens of thousands of genes, which can be modeled as true gene expressions mixed with noise in data generation, variability of the underlying biological systems etc. We develop a novel BN structure learning algorithm for reconstructing GRNs.

The second application is to develop a BN method for protein-protein interaction (PPI) assessment. PPIs are the foundation of most biological mechanisms, which provides one of the most valuable resources from which annotations of genes and proteins can be discovered. Experimentally, recently-developed high-throughput technologies have been carried out to reveal protein interactions in many organisms. However, high-throughput interaction data often contain a large

number of spurious interactions. In this thesis, I develop a novel in silico model for PPI assessment. Our model is based on a BN that integrates heterogeneous data sources from different organisms. The results show that it outperforms naïve Bayes and a manual constructed Bayesian Network.

My work in this thesis can be summarized in the following aspects:

Dynamic Bayesian network learning

In this thesis, first I propose a fast method to learn dynamic Bayesian network structure preferable to model complex biological networks with feedback loops. The current DBN methods may limit their applications in large-scale network analysis due to relatively low accuracy of regulatory network prediction and excessive computation time. Our method uses differential mutual information to select a set of potential regulators for each gene first, which alleviate the two problems associated with standard DBNs. In this case, we employ mutual information to measure the dependence between two nodes at different time points, thus we name it differential mutual information.

This method is tested on two simulated networks and a real biological pathway. Our experimental results show that the proposed method produces better overall performance than the commonly-used BDe, ML, and BIC methods. Furthermore, our method is much faster than BIC and ML methods, especially for networks of moderate size. This is attributed to the fact that we only evaluate a limited number of potential parents (selected by our DMI) for each node. Thus, our method can be applied to very large networks learning where standard DBN is infeasible. Experimental results carried out on the cell cycle pathway of Saccharomyces cerevisiae further demonstrate the effectiveness of the proposed method. We predicted some new interactions that were not included in the original KEGG pathways, but verified by biological experiments elsewhere.

Bayesian network learning using Markov blanket

Markov blanket (MB) of a node in a Bayesian network consist its parents, children and children's parents. MB contains all the informations about the node, in other words, given its MB, the node is independent to all the other nodes outside the MB. By definition, Markov blanket of a variable is equivalent to its strongly relevant feature set. Thus, applying existing feature selection techniques, we can identify Markov blanket of a target. The knowledge of MB of a node in a

Bayesian network can greatly simplify the recovery of its local structure.

In this thesis, I propose a MB based structure learning method by first applying feature selection to identify MB of each nodes, then examining the constrains represented in correlations among close nodes to further recover local structures, then the whole network. The method is tested on a golden network, i.e., RAF pathway. The result shows that our method out-perform Hillclimbing and MCMC, and also reveal new causal relationships confirmed in literature.

Using Markov blanket for feature selection in classification

I theoretically prove an asymptotic property of K2 learning algorithm, that is, when sample size is very large, K2 in random ordering can always detect a edge if and only if it is between two adjacent nodes in the real structure. This theorem can be used to find all undirected edges in the graph. This precess can be fast due to the efficiency of K2. Then limited correlation tests among local nodes can quickly uncover the Markov blanket of the class variable in the classification scenario.

This method is evaluated on a golden structure- the Alarm network. The results show that our method obtains more accurate Markov blankets than the representative MB learning algorithm (KS algorithm) proposed by Koller and Thrun. Furthermore, when the learned MB used as feature set for classification, our method also out-perform KS algorithm on the benchmark Molecular dataset.

Classification orientated Bayesian network: a tree- augmented naïve Bayesian (TAN)

Standard Bayesian scoring functions are designed for structure learning, not particularly for classification tasks. The scoring functions measure the errors of the learned Bayesian network over all the variables in the domain, hence, minimizing this error does not necessarily minimize the local error in predicting the class variable given the attributes, especially when there are many attributes. We introduce a classification aimed Bayesian network-TAN (Tree Augmented Naive Bayes), which has the computational simplicity and functional robustness.

In this thesis, TAN is used for protein-protein interaction (PPI) assessment. PPIs are the foundation of most biological mechanisms, which provides one of the most valuable resources from which annotations of genes and proteins can be discovered. Experimentally, recently-developed

high-throughput technologies have been carried out to reveal protein interactions in many organisms. However, high-throughput interaction data often contain a large number of spurious interactions. We introduce the learning of a TAN (Tree Augmented Naive Bayes) based network, which integrates heterogeneous data sources from model organisms to assess PPI in a target organism. The empirical results demonstrate that the TAN model outperforms naïve Bayes and a manual constructed Bayesian Network; furthermore, the model convinces that sufficient model organisms can provide accurate PPI prediction.

7.2 Future works: Learning a Bayesian network with incomplete data

In this thesis, I assume data are complete, i.e., no missing values exist. However, the issue of incomplete data is not rare in biological data analysis. For example, in my study of PPI prediction, nearly one fourth of instances have missing values. For the seek of simplicity, the contribution from missing attributes is ignore in this thesis.

Missing data happen in two ways: at random or not at random. Data missing at random means the missing is not related to the states of the other variables, such as the recording problems or some similar error. This type of missing is relative easy to handle, and the standard practice is to apply *Expectation and Maximization* (EM) algorithm proposed by Dempster et. al [144] and McLachlan and Krishnam [145]. If data missing does not occur at random, other variables contribute to the situation. For example, in a drug study a patient may become too sick to complete the study, due to a side effect of the drug. So the fact that the result variable is missing depends directly on the value of the side effect variable. To this more complicate situation, Cooper [146] and Spirtes et.al [147] suggested some solutions.

Bayesian networks have the advantage when facing missing values, due to the probability presentation of the dependence relationship among variables. I am interested in studying Bayesian network learning in the presence of missing data in the future.

A. Proof of an asymptotic property of K2 algorithm

Lemma: Assume the number of cases in the database $\mathscr{D}$ generated from a Bayesian belief network $\mathscr{G}$ with positive conditional probabilities, approaches infinity. If a pair of nodes **a** and **b** are adjacent in $\mathscr{G}$, then adding the front node in any node ordering to the parent set of the other node can always increase the likelihood in $K2$ metrics.

Proof

Without loss of generality, assume **a** is in front of **b** in a given node ordering $\mathscr{O}$, modify a structure B_s, in which **a** is not in the parent set of **b** ($\mathbf{a} \notin \pi(\mathbf{b}) = Z$), by adding **a** as a parent of **b** to form a new structure B_s' with the new parent set $\pi'(\mathbf{b}) = Z \cup \{\mathbf{a}\}$, we need to prove the likelihood increases, i.e.,

$$P(\mathscr{D}|B_s) < P(\mathscr{D}|B_s') \tag{A.1}$$

In $K2$ metrics, likelihood can be computed as follows,

$$P(\mathscr{D}|B_s) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \tag{A.2}$$

Where,

n: number of nodes in B_s

q_i: number of unique instantiations of π_i (set of parents of variable X_i

r_i: number of possible value assignments of X_i

N_{ijk}: number of cases when X_i has the k^{th} value and π_i has the j^{th} value

$N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$

Comparing B_s and B_s' , the only change is at node **b**. Applying (A.2) to (A.1) and only consid-

ering **b**, we need to prove,

$$\prod_{q=1}^{r_Z} \frac{(r_b-1)!}{(N_{Z_q}+r_b-1)!} \prod_{k=1}^{r_b} N_{Z_q b_k}! < \prod_{m=1}^{r_a} \prod_{q=1}^{r_Z} \frac{(r_b-1)!}{(N_{Z_q a_m}+r_b-1)!} \prod_{k=1}^{r_b} N_{Z_q a_m b_k}! \tag{A.3}$$

Where,

r_a, r_b and r_Z: the numbers of possible assignments for variable **a**, **b** and Z respectively;

N_{Z_q}: number of cases when Z has the q^{th} value

$N_{Z_q b_k}$: number of cases when Z and **b** have their q^{th} and k^{th} values respectively

$N_{Z_q a_m}$: number of cases when Z and **a** have their q^{th} and m^{th} values respectively

$N_{Z_q a_m b_k}$: number of cases when Z, **a** and **b** have their q^{th}, m^{th} and k^{th} values respectively

N: total number of cases in (D)

$$N_{Z_q} = \sum_{m=1}^{r_a} N_{Z_q a_m} = \sum_{k=1}^{r_b} N_{Z_q b_k}$$

$$N = \sum_{q=1}^{r_Z} N_{Z_q} = \sum_{q=1}^{r_Z} \sum_{m=1}^{r_a} N_{Z_q a_m} = \sum_{q=1}^{r_Z} \sum_{k=1}^{r_b} N_{Z_q b_k} = \sum_{q=1}^{r_Z} \sum_{m=1}^{r_a} \sum_{k=1}^{r_b} N_{Z_q a_m b_k}$$

Since the conditional probabilities of $\mathscr{G}$ are positive, all states of the variables are possible to be instantiated. When N is approaching infinity, all terms describing numbers of cases in (A.3) are large. Thus Stirling Approximation can be used on those terms. Stirling Approximation states,

$$n! \approx \sqrt{2\pi n}(n/e)^n, \text{when } n \text{ is large} \tag{A.4}$$

Denote the left side of (A.3) as S_1, and apply Stirling Approximation to S_1:

$$s_1 \approx \prod_{q=1}^{r_Z} \frac{(r_b-1)!}{\sqrt{2\pi(N_{Z_q}+r_b-1)}((N_{Z_q}+r_b-1)/e)^{(N_{Z_q}+r_b-1)}} \prod_{k=1}^{r_b} \sqrt{2\pi N_{Z_q b_k}}(N_{Z_q b_k}/e)^{N_{Z_q b_k}} \tag{A.5}$$

Take the logarithm on both sides,

$$\begin{aligned}\log S_1 &\approx r_Z \log[(r_b-1)!(\sqrt{2\pi}e)^{r_b-1}] + \\ &\sum_{q=1}^{r_Z}[-\frac{1}{2}(r_b-1)\log(N_{Z_q}+r_b-1)+\sum_{k=1}^{r_b}(N_{Z_qb_k}+1/2)\log\frac{N_{Z_qb_k}}{N_{Z_q}+r_q-1}] \quad \text{(A.6)}\end{aligned}$$

On the right of (A.6), the first term is a constant, and inside the summation over q, the first term is in order of $\log N$ and the other term over k in order of N is dominant.

Thus, only keep the dominant terms,

$$\begin{aligned}\log S_1 &\approx \sum_{q=1}^{r_Z}\sum_{k=1}^{r_b}(N_{Z_qb_k}+\frac{1}{2})\log\frac{N_{Z_qb_k}}{N_{Z_q}+r_b-1} \\ &\approx \sum_{q=1}^{r_Z}\sum_{k=1}^{r_b}N_{Z_qb_k}\log\frac{N_{Z_qb_k}}{N_{Z_q}} \\ &= N\sum_{q=1}^{r_Z}\sum_{k=1}^{r_b}\frac{N_{Z_qb_k}}{N}\log\frac{N_{Z_qb_k}}{N_{Z_q}} \\ &= N\sum_{q=1}^{r_Z}\sum_{k=1}^{r_b}P(Z=Z_q,b=b_k)\log P(b=b_k|Z=Z_q) \\ &= -N\cdot H(b|Z) \quad \text{(A.7)}\end{aligned}$$

Where,

$P(Z=Z_q,b=b_k)=\frac{N_{Z_qb_k}}{N}$: joint probability of $Z=Z_q$, and $b=b_k$

$P(b=b_k|Z=Z_q)=\frac{N_{Z_qb_k}}{N_{Z_q}}$: conditional probability of $b=b_k$, giving $Z=Z_q$

$H(b|Z)$: entropy of $\mathbf{b}$, conditioning on Z.

Let the right side of (A.3) as S_2, and apply Stirling Approximation (A.4) to S_2,

$$s_2 \approx \sum_{m=1}^{r_a}\sum_{q=1}^{r_Z}\frac{(r_b-1)!}{\sqrt{2\pi(N_{Z_qa_m}+r_b-1)}\left(\frac{N_{Z_qa_m}+r_b-1}{e}\right)^{N_{Z_qa_m}+r_b-1}}\sum_{k=1}^{r_b}\sqrt{2\pi N_{Z_qa_mb_k}}\left(\frac{N_{Z_qa_mb_k}}{e}\right)^{N_{Z_qa_mb_k}} \quad \text{(A.8)}$$

Take the logarithm on both sides,

$$\begin{aligned}\log S_2 &\approx r_a r_Z \log[(r_b-1)!(\sqrt{2\pi}e)^{r_b-1}] + \\ &\sum_{m=1}^{r_a}\sum_{q=1}^{r_Z}[-\frac{1}{2}(r_b-1)\log(N_{Z_q a_m}+r_b-1)+ \\ &\sum_{k=1}^{r_b}(N_{Z_q a_m b_k}+\frac{1}{2})\log\frac{N_{Z_q a_m b_k}}{N_{Z_q a_m}+r_b-1}] \end{aligned} \tag{A.9}$$

On the right side of (A.9), the first term is a constant, and inside the summation over m and q, the first term is in order of $\log N$ and the second term is in the order of N, which is dominant.

$$\begin{aligned}\log S_2 &\approx \sum_{m=1}^{r_a}\sum_{q=1}^{r_Z}\sum_{k=1}^{r_b}(N_{Z_q a_m b_k}+\frac{1}{2})\log\frac{N_{Z_q a_m b_k}}{N_{Z_q a_m}+r_b-1} \\ &\approx \sum_{m=1}^{r_a}\sum_{q=1}^{r_Z}\sum_{k=1}^{r_b}N_{Z_q a_m b_k}\log\frac{N_{Z_q a_m b_k}}{N_{Z_q a_m}} \\ &= N\sum_{m=1}^{r_a}\sum_{q=1}^{r_Z}\sum_{k=1}^{r_b}\frac{N_{Z_q a_m b_k}}{N}\log\frac{N_{Z_q a_m b_k}}{N_{Z_q a_m}} \\ &= N\sum_{m=1}^{r_a}\sum_{q=1}^{r_Z}\sum_{k=1}^{r_b}P(Z=Z_q,a=a_m,b=b_k)\log P(b=b_k|Z=Z_q,a=a_m) \\ &= -N\cdot H(b|Z,a) \end{aligned} \tag{A.10}$$

Where,

$P(Z=Z_q,a=a_m,b=b_k)=\dfrac{N_{Z_q a_m b_k}}{N}$: joint probability of $Z=Z_q,a=a_m,$ and $b=b_k$

$P(b=b_k|Z=Z_q,a=a_m)=\dfrac{N_{Z_q a_m b_k}}{N_{Z_q a_m}}$: conditional probability of $b=b_k$, giving $Z=Z_q,a=a_m$

$H(b|Z,a)$: the entropy of **b** conditioning on **a** and Z.

Subtracting (A.10) from (A.7),

$$\begin{aligned} \log S_1 - \log S_2 &\approx -N \cdot H(b|Z) - (-N \cdot H(b|Z,a)) \\ &= -N \cdot I(b|Z,a) \end{aligned} \tag{A.11}$$

Where,

$I(b|Z,a)$: information gain of **b** conditioning on Z and **a**.

Since **a** and **b** b adjacent in G as given, **a** is either a parent or a child of **b**. In other words, **a** is in the Markov Blanket of **b**, i.e., $a \in Mb(b)$. Therefore, **a** is informative for **b**, giving all the other variables, which means for any Z, after adding in **a**, the information gain is always positive, i.e.,

$$I(b|Z,a) > 0 \tag{A.12}$$

Substitute (A.12) into (A.11),

$\log S_1 - \log S_2 < 0$ or $S_1 < S_2$

The Lemma (A.1 is proven.

Theorem: Assume the number of cases in the database $\mathscr{D}$ generated from a Bayesian belief network $\mathscr{G}$ with positive conditional probabilities, approaches infinity. An edge between two nodes can be identified by $K2$ algorithm in all node ordering if and only if these two nodes are adjacent in $\mathscr{G}$.

Proof

First, briefly recall the $K2$ algorithm. $K2$ algorithm starts the model selection process with a structure with no arcs, then forward constructs parent sets of each node i by adding the node from available parent candidates (nodes before node i in the given node ordering), which increases the joint probability $P(B_S, \mathscr{D})$ most. The process terminates when no expansion of parent sets can increase $P(B_S, \mathscr{D})$ further or when the limit of maximum number of parents has been reached. Therefore, for a given node ordering and given constrain of maximum number of parents, $K2$ returns the model $B_{S,max}$, which maximizes the joint probability, i.e.,

$$B_{S,max} = \underset{B_S}{\operatorname{argmax}} P(B_S, \mathscr{D}) = \underset{B_S}{\operatorname{argmax}} P(\mathscr{D}|B_S)P(B_S) \tag{A.13}$$

If no prior knowledge is available, often uniform distribution over B_S is assumed, i.e., $P(B_S)$ is a constant. Thus,

$$B_{S,max} = \underset{B_S}{\operatorname{argmax}} P(B_S, \mathscr{D}) = \underset{B_S}{\operatorname{argmax}} P(\mathscr{D}|B_S) \tag{A.14}$$

The best model is the one with the maximum likelihood under the given constrain.

Part I: Prove the statement that if nodes **a** and **b** are adjacent in $\mathscr{G}$, then either $a \in \pi(b)$) or $b \in \pi(a)$ in any given node ordering, in other words, edge between **a** and **b** can always be identified regardless the given node ordering. Without loss of generality, assume **a** is in front of **b** in a given ordering in the following proof. Following the preceding Lemma, since **a** and **b** are adjacent in $\mathscr{G}$, for any given node ordering, adding **a** into $\pi(b)$ can always increases the likelihood. If the maximum number of parents is large, the node **a** will be added in $\pi(b)$ before $K2$ terminates eventually.

Part II: Prove the statement that if **a** and **b** are not adjacent in $\mathscr{G}$, then there exists a node ordering $\mathscr{O}_0$ (without loss of generality, assume **a** is before **b** in $\mathscr{O}_0$, such that in the returned structure from $K2$, B_{S_0}, $a \notin \pi_0(b)$. Since **a** and **b** are not adjacent in $\mathscr{G}$, there are two cases:

Case 1: **a** is not in the Markov blanket of **b**, i.e., $a \notin Mb(b)$.

Case 2: $a \in Mb(b)$, but **a** and**b** are spouse relationship through child node c, i.e., **a** and **b** are related only when c is given.

Let $\mathscr{A}$ be the set of all the nodes before **b**.

Case 1: $a \notin Mb(b)$, Let $\mathscr{O}_0$ be a node ordering with $Mb(b) \subseteq \mathscr{A}$. Based on equation (A.7), the likelihood for the structure B_{S_0} associated with node **b**, S_0 can be approximated as:

$$\log S_0 \approx -N \cdot H(b|\pi_0(b)) \tag{A.15}$$

Thus, having $\pi_0(b)$ as the smallest set which has the lowest entropy under the given ordering,

$$H(b|\pi_0(b)) = H(b|\mathscr{A}) = H(b|Mb(b), \mathscr{A} \backslash Mb(b)) = H(b|Mb(b)) \tag{A.16}$$

As Markov blanket of a node is unique and the smallest set which block the node from all the other nodes, we can conclude that $\pi_0(b)$ and $Mb(b)$ are identical. Hence, $a \notin \pi_0(b)$.

Case 2: **a** and **b** are related only when c is given. Let $\mathscr{O}_0$be a node ordering with $c \notin \mathscr{A}$. Since $\pi_0 \subseteq \mathscr{A}, c \notin \pi_0(b)$. Assuming $a \in \pi_0(b)$,

$$H(b|\pi_0(b)\backslash, a) = H(b|\pi_0(b)\backslash a) = H(b|\pi_0(b)) \tag{A.17}$$

(A.17) is a controversy to the fact that $\pi_0(b)$ should be the smallest set which has the lowest entropy under the given ordering. Therefore, the assumption $a \in \pi_0(b))$ can't be correct, which proves $a \notin \pi_0(b)$. Thus, in all cases when **a** and **b** are not adjacent in $\mathscr{G}$, we can always find a node ordering such that in the structure returned by $K2$, there is no edge between **a** and **b**.

Bibliography

[1] J. M. Roberts, T. A. Parlikar, T. Heldt, and G. C. Verghese, "Bayesian Networks for Cardiovascular Monitoring," In *Proceedings of the 28th IEEE EMBS Annual International Conference*, pp. 205–209 (New York City, USA, 2006).

[2] P. J. F. Lucas, L. C. van der Gaag, and A. Abu-Hanna, "Bayesian networks in biomedicine and health-care," Artifical Intelligence in Medicine **30,** 201–214 (2004).

[3] G. Rutledge, G. Thomsen, I. Beinlich, B. Farr, B. Sheiner, and L. Fagan, *Proceedings of the Thirteenth Annual Symposium on Computer Applications in Medical Care* (Washington, D.C., 1989), pp. 315–319.

[4] W. Long, "Temporal reasoning for diagnosis in a causal probabilistic knowledge base," Artifical Intelligence in Medicine **8,** 193–215 (1996).

[5] D. Nikovski, "Constructing Bayesian Networks for Medical Diagnosis from Incomplete and Partially Correct Statistics," IEEE Trans. Knowledge and Data Engineering **12,** 509–516 (2000).

[6] C. Berzuini, R. Bellazzi, S. Quaglini, and D. J. Spiegelhalter, "Bayesian networks for patient monitoring," Artifical Intelligence in Medicine **4,** 243–260 (1992).

[7] I. Maglogiannis, E. Zafiropoulos, A. Platis, and C. Lambrinoudakis, "Risk Analysis of a Patient Monitoring System Using Bayesian Network Modeling," Journal of Biomedical Informatics **39(6),** 637–647 (2006).

[8] O. Pietquin and T. Dutoit, "A Probabilistic Framework for Dialog Simulation and Optimal Strategy Learning," IEEE Trans. Speech and Audio Processing **14(2),** 589–599 (2005).

[9] J. Agosta, in *Uncertainty in Artificial Intelligence*, T. L. R. Shacter, L. N. Kanal, and J. Lemmer, eds., (North-Holland, New York, 1990), Vol. 4, pp. 397–405.

[10] D. Heckerman, "Probabilistic Similarity Networks," Networks **20,** 607–636 (1990).

[11] N. Friedman and et al., "Using Bayesian Networks to Analyze Expression Data," Data. J. Comput. Biol. **7,** 601–620 (2000).

[12] D. Koller and N. Friedman, *Probabilistic Graphical Models - Principles and Techniques* (The MIT Press, Cambridge, massachusetts, London, England, 2009).

[13] E. H. Herskovits and G. F. Cooper, *Proceedings of the Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-90)* (Elsevier Science, New York, NY, 1990), pp. 54–63.

[14] J. Pearl, *Causality: Models, Reasoning, and Inference* (Cambridge University Press, New York, NY, USA, 2000).

[15] R. E. Neapolitan, *Learning Bayesian Networks* (Pearson Education, Inc., Pearson Prentice Hall, Upper Saddle River, NJ., 2004).

[16] N. Wermuth and S. Lauritzen, "Graphical and Recursive Models for Contingence Tables," Biometrika **72,** 537–552 (1983).

[17] D. Geiger, A. Paz, and J. Pearl, *Proc. AAAI-90* (1990), pp. 770–776.

[18] P. Spirtes, C. Glymour, and R. Scheines, "Causation, Prediction and Search,", 1993.

[19] C. Meek, "Causal Inference and Causal Explanation with Background Knowledge," Proceedings of the eleventh international conference on uncertainty in artificial intelligence (1995).

[20] L. de Campos and I. Huete, "On the Use of Independence Relationships for Learning Simplified Brief Networks," International Journal of Intelligent Systems **12,** 495–522 (1997).

[21] L. de Campos, "Independency Relationships and Learning Algorithms for Singly Connected Networks," Journal of Experimental and Theoretical Artificial Intelligence **10,** 511–549 (1998).

[22] L. de Campos and J. Huete, "A New Approach for Learning Belief Networks Using Independence Criteria," International Journal of Approximate Reasoning 24 (2000).

[23] J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu, "Learning Bayesian Networks from data: an Information-theory based Approach," Artificial Intelligence **137(1-2),** 43–90 (2002).

[24] G. F. Cooper and E. Herskovits, "A Bayesian Method for the Induction of Probabilistic Networks from Data," Machine Learning **9,** 309–347 (1992).

[25] R. R. Bouckaert, "Probabilistic Network Construction Using the Minimum Description Length Principle,", 1993.

[26] J. Suzuki, "A construction of Bayesian networks from databases based on MDL scheme," In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, D. Heckerman and A. Mamdani, eds., pp. 266–273 (Morgan Kaufmann, San Francisco, 1993).

[27] W. Lam and F. Bacchus, "Learning Bayesian belief networks-An approach based on the MDL principle," Computational Intelligence 10 (1994).

[28] D. M. Chickering, "A transformational characterization of Bayesian network structures," In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, S. Hanks and P. Besnard, eds., pp. 87–98 (Morgan Kaufmann, 1995).

[29] D. Heckerman and D. Geiger, "Likelihoods and Parameter Priors for Bayesian Networks," Technical Report No. MSR-TR-95-94, Microsoft Research (1995) .

[30] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: the combination of knowledge and statistical data," Machine Learning **20(3),** 197–243 (1995).

[31] N. Friedman and D. Koller, "Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks," Machine Learning **50,** 95–125 (2003).

[32] D. M. Chickering, "Learning Bayesian networks is NP-Complete," in *Learning from Data: Artificial Intelligence and Statistics*, D. Fisher and H. Lenz, eds., (Springer-Verlag, 1996), Vol. V, pp. 121–130.

[33] D. M. Chickering, D. Heckerman, and C. Meek, "Large-Sample learning of Bayesian networks is NP-Hard," Journal of Machine Learning Research **5,** 1287–1330 (2004).

[34] D. Madigan and J. York, "Bayesian graphical models for discrete data," International Statistics Review **63,** 215–232 (1995).

[35] P. Larranaga and et al., "Learning Bayesian network structures by searching for the best ordering with genetic algorithms," IEEE Trans. Systems, Man, and Cybernetic **26,** 487–493 (1996).

[36] P. Larranaga and et al., "Structure learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters," IEEE Trans. Pattern Analysis and Machine Intelligence **18,** 912–926 (1996).

[37] D. Chickering, D. Geiger, and D. Heckerman, *Preliminary Papers 5th International Workshop Artificial Intelligence and Statistics* (1995).

[38] P. Muntenau and D. Cau, "Efficient score-based learning of equivalence classes of Bayesian networks," in *Lecture Notes in Artificial Intelligence* (Springer, 2000), pp. 96–105.

[39] J. Tian, "A branch-and-bound algorithm for MDL learning Bayesian network," In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 580–587 (2000).

[40] M. Singh and M. Valtorta, "An algorithm for the construction of Bayesian network structures from data," In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pp. 259–365 (1993).

[41] M. Singh and M. Valtorta, "Construction of Bayesian network structures from data: a brief survey and an effective algorithm," International Journal of Approximate Reasoning **12,** 111–131 (1995).

[42] P. Spirtes and C. Meek, "Learning Bayesian networks with discrete variables from data," In *Proceedings of the First International conference on Knowledge Discovery and Data Mining*, 294-299, ed., (1995).

[43] D. Dash and M. Druzdzel, "A hybrid anytime algorithm for the construction of causal models from sparse data," In *Proceedings of the Fifteenth Conference on Uncertainty on Artificial Intelligence*, pp. 142–149 (1999).

[44] S. Acid and L. M. de Campos, "Learning right sized belief networks by means of a hybrid methodology," in *Lecture Notes on Artificial Intelligence* (Springer, 2000), No. 1910, pp. 309–315.

[45] S. Acid and L. M. de Campos, "A hybrid methodology for learning belief networks: Benedict," International Journal of Approximate Reasoning **2,** 235–262 (2001).

[46] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," Machine Learning **65,** 31–78 (2006).

[47] S. Imoto, T. Goto, and S. Miyano, "Estimation of genetic networks and functional structures between genes by using Bayesian network and nonparametric regression," Pacific Symposium on Biocomputing (PSB02) **7,** 175–186 (2002).

[48] X. Chen, G. Anantha, and X. Wang, "An effective structure learning method for constructing gene networks," Bioinformatics **22,** 1367–1374 (2006).

[49] L. Badea, "Determining the direction of causal influence in large probabilistic networks: a constraint-based approach," In *Proceedings of the 16th European Conference on Artificial Intelligence*, pp. 263–267 (IOS Press, Valencia, Spain, 2004).

[50] A. Bernard and A. J. Hartemink, "Informative structure priors: joint learning of dynamic regulatory networks from multiple types of data," Pac. Symp. Biocomput pp. 459–470 (2005).

[51] N. Friedman and M. Goldzmidt, "Learning Bayesian networks with local structure," In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, pp. 201–210 (1996).

[52] A. J. Hartemink, D. K. Jaakkola, and R. A. Young, "Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks," Pac. Symp. Biocomput pp. 422–433 (2001).

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

[54] S. Otta, S. Imoto, and S. Miyano, "Finding optimal models for small gene networks," Pac. Symp. Biocomput pp. 555–567 (2004).

[55] J. M. Pena, J. Bjorkegren, and J. Tegner, "Growing Bayesian network models of gene networks from seed genes," Bioinformatics **21, suppl. 2,** ii224–ii229 (2005).

[56] M. Zou and D. Conzen, "A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data," Bioinformatics **21(1),** 71–79 (2005).

[57] 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 **308,** 523–529 (2005).

[58] B. Han and X. Chen, "bNEAT: a Bayesian network method for detecting epistatic interactions in genome-wide association studies," BMC Genomics **12(suppl 2),** S9 (2011).

[59] J. R. Bradford, C. J. Needham, A. J. Bulpitt, and D. R. Weathead, "Insights into protein-protein interfaces using a Bayesian network prediction method," J. Mol. Biol. **362,** 365–386 (2006).

[60] X. Lin, M. Liu, and X. Chen, "Assessing reliability of protein-protein interactions by integrative analysis of data in model organisms," BMC Bioinformatics **10(Suppl 4),** S5 (2009).

[61] D. Haussler, "Decision theoretic generalizations of the PAC model for neural net and other learning applications," Inform. Control **100(1),** 78–150 (1992).

[62] L. Hartwell, J. Hopfield, S. Leibler, and A. Murray, "From molecular to modular cell biology," Nature **402,** C47–C52 (1999).

[63] J. Hasty, D. McMillen, and J. J. Collins, "Engineered gene circuits," Nature **420,** 224–230 (2002).

[64] R. Lipshutz, D. Morris, M. Chee, E. Hubbell, M. Kozal, N. Shah, N. Shen, R. Yang, and S. Fodor, "Using oligonucleotide probe arrays to access genetic diversity," Biotechniques **19,** 442–447 (1995).

[65] M. Schena, D. Shalon, R. Davis, and P. Brown, "Quantitative monitoring of gene expression patterns with a complementary DNA microarray," Science **270,** 467–470 (1995).

[66] N. Friedman, M. Goldszmidt, and A. Wyner, *Proc Fifteenth Conf on Uncertainty in Artificial Intelligence (UAI)* (Morgan Kaufmann, Stockholm, Sweden, 1999), pp. 196–205.

[67] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, *Proceedings of the 4th international conference on Computational molecular biology* (2000), No. 127-135.

[68] D. Pe'er, A. Rege, G. Elidan, and N. Friedman, "Inferring subnetworks from perturbed expression profiles," Bioinformatics **17,** S215–224 (2001).

[69] N. Friedman, K. Murphy, and S. Russwell, *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann, Madison, WI, 1998), No. 139-147.

[70] K. Murphy and S. Mian, "Modeling gene expression data using dynamic Bayesian networks," Technical report, Computer Science Division, University of California, Berkeley, CA (1999) .

[71] I. Ong, J. Glasner, and D. Page, "Modeling regulatory pathways in E. coli from time series expression profiles," Bioinformatics **18,** S241–248 (2002).

[72] B. Perrin, L. Ralaivola, A. Mazurie, S. Bottani, J. Mallet, and F. d'Alche Buc, "Gene networks inference using dynamic Bayesian networks," Bioinformatics **19,** ii138–148 (2003).

[73] S. Kim, S. Imoto, and S. Miyano, "Dynamic Bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data," Biosystems **75,** 57–65 (2004).

[74] J. Proakis, *Digital Communications* (Mcgraw – Hill, New York, 2001).

[75] P. Spellman, G. Sherlock, M. Zhang, V. Iyer, K. Ander, M. Eisen, D. B. P. Brown, and D. Futcher., "Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast Saccharomyces cerevisiae by Microarray Hybridization.," Molecular Biology of the Cell **9,** 3273–3297 (1998).

[76] R. Basco, M. Segal, and S. Reed, "Negative regulation of G1 and G2 by S-phase cyclones of Saccharomyces cerevisiae," Molecular and Cellular Biology **15,** 5030– 5042 (1995).

[77] H. Wijnen and B. Futcher, "Genetic analysis of the shared role of CLN3 and BCK2 at the G1-S Transition in Saccharomyces cerevisiae," Genetics **152,** 1131–1143 (1999).

[78] D. Margaritis and S. Thrun, in *Advances in Neural Information Processing Systems, NIPS*, S. A. Solla, T. K. Leen, and K. Muller, eds., (2000), Vol. 12, pp. 501–511.

[79] I. Tsamardinos, C. F. Aliferis, and A. Statnikov, *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM Press, 2003), pp. 673–678.

[80] J. P. Pellet and A. Elisseeff, "Using Markov blanket for causal structure learning," Journal of Machine Learning Research **9,** 1295–1342 (2008).

[81] R. G. Cowell, A. P. Dawid, and et. al, *Probabilistic Networks and Expert Systems* (Springer-Verlag, New York, 1999).

[82] S. L. Lauritzen and D. J. Spiegelhalter, "Local computations with probabilities on graphical structures and their application to expert systems," Journal of the Royal Statistical Society **B 50,** 157–224 .

[83] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference* (Morgan Kaufmann, 1988).

[84] P. Spirtes, *Causation, Prediction, and Search*, 2nd ed. (MIT Press, 2001).

[85] W. Kolch, "Meaningful relationships: the regulation of the Ras/Raf/MEK/ERK pathway by protein interactions," Biochem J. **351,** 289–305 (2000).

[86] A. V. Werhli, M. Grzegorczyk, and D. Husmeier, "Comparative evaluation of reverse engineering gene regulatory networks with relevance networks, graphical gaussian models and bayesian networks," Bioinformatics **22,** 2523–2531 (2006).

[87] M. Schilstra, "The NetBuilder Home page,", 2002.

[88] "The Moments of the Z and F Distributions," Biometrika **36,** 394–403 (1949).

[89] M. G. Madden, "Evaluation of the performance of the markov blanket Bayesian classifier algorithm," Technical Report No. NUIG-IT-011002, Department of Information Technology, National University of Ireland, Galway, Ireland (2002) .

[90] D. Koller and M. Sahami, *Proceedings of the Thirteenth International Conference on Machine Learning* (1996).

[91] E. P. Xing, M. J. Jordan, and R. M. Karp, *ICML'01 Proceedings of the Eighteenth International Conference on Machine Learning* (2001), pp. 601–608.

[92] S. Kullback and R. A. Leibler, "On Information and Sufficiency," Annals of Mathematical Statistics **22,** 79–86 (1951).

[93] S. Kullback, *Information theory and statistics* (John Wiley and Sons, New York, 1959).

[94] N. Friedman and M. Goldszmidt, *Proc. AAAI-96* (1996), Vol. 2, pp. 1277–1284.

[95] C. Harley and R. Reynolds, "Analysis of E. Coli Promoter Sequences," Nucleic Acids Research 15 (1987).

[96] C. Royer, in *Protein-protein interactions* (the Biophysics Textbook Online (BTOL), 1999).

[97] P. M. Kim, L. J. Lu, Y. Xia, and M. B. Gerstein, "Relating three-dimensional structures to protein networks provides evolutionary insights," Science **314,** 1938–41 (2006).

[98] J. Rachlin, D. D. Cohen, C. Cantor, and S. Kasif, "Biological context networks: a mosaic view of the interactome"," Mol Syst Biol 2 (2006).

[99] G. T. Hart, A. K. Ramani, and E. M. Marcotte, "How complete are current yeast and human protein-interaction networks?," Genome Biology 7 (2006).

[100] R. M. Kini and J. H. Evans, "Prediction of potential protein-protein interaction sites from amino acid sequence. Identification of a fibrin polymerization site," FEBS Lett. **385,** 81–86 (1996).

[101] S. Jones and J. M. Thornton, "Prediction of protein-protein interaction sites using patch analysis," J. Mol. Biol. **272,** 133–143 (1997).

[102] J. R. Bock and D. A. Gough, "Predicting protein-protein interactions from primary structure," Bioinformatics **17,** 455–460 (2001).

[103] F. Pazos, M. Helmer-Citterich, G. Ausiello, and A. Valencia, "Correlated mutation contain information about protein-protein interaction," J. Mol. Biol. **1,** 511–523 (1997).

[104] T. Dandekar, B. Snel, M. Huynen, and P. Bork, "Conservation of gene order: a fingerprint of proteins that physically interact," Trends Biochem. Sci. **23,** 324–328 (1998).

[105] A. J. Enright, I. Iliopoulos, N. C. Kyrpides, and C. A. Ouzounis, "Protein interaction maps for complete genomes based on gene fusion events," Nature **402,** 25–26 (1999).

[106] E. M. Marcotte, M. Pellegrini, H. L. Ng, D. W. Rice, T. O. Yeastes, and D. Eisenberg, "Detecting protein function and protein-protein interactions from genome sequences," Science **285,** 751–753 (1999).

[107] M. Huynen, B. Snel, W. Lathe, and P. Bork, "Predicting protein function by genomic context: quantitative evaluation and qualitative inferences," Genome research **10,** 1204–1210 (2000).

[108] C. S. Goh, A. A. Bogan, M. Joachimiak, D. Walther, and F. E. Cohen, "Co-evolution of proteins with their interaction partners," J. Mol. Biol. **299,** 283–293 (2000).

[109] F. Pazos and A. Valencia, "Similarity of phylogenetic trees as indicator of protein-protein interaction," Protein Eng. **14,** 609–614 (2001).

[110] A. K. Ramani and E. M. Marcotte, "Exploiting the co-evolution of interacting proteins to discover interaction specificity," J. Mol. Biol. **327,** 273–284 (2003).

[111] E. Sprinzak and H. Margalit, "Correlated sequence-signatures as markers of protein-protein interactions," J. Mol. Biol. **311,** 681–692 (2001).

[112] W. K. Kim, J. Park, and J. K. Suh, "Large scale statistical prediction of protein-protein interaction by potentially interacting domain (PID) pair," Genome Informatics **13,** 42–50 (2002).

[113] S. Ng, Z. Zhang, and S. Tan, "Integrative approach for computationally inferring protein domain interactions," Bioinformatics **10,** 359–365 (2003).

[114] X. Chen and M. Liu, "Prediction of protein-protein interactions using random decision forest framework," Bioinformatics **21,** 4394–4400 (2005).

[115] L. Zhang, L. Wong, O. King, and F. Poth, "Predicting co-complexed protein pairs using genomic and protenomic data integration," BMC Bioinformatics 5 (2004).

[116] W. Zhong and P. Sternberg, "Genome-wide Prediction of C. elegans Genetic Interactions," Science **311,** 1481–1484 (2006).

[117] C. Myers, D. Robson, A. Wible, M. Hibbs, C. Chiriac, C. Theesfeld, K. DOlinski, and O. Troyanskaya, "Discovery of biological networks from diverse functional genomic data," Genome Biology **6,** R114 (2005).

[118] R. Jansen, H. Yu, D. Greenbaum, Y. Kluger, N. J. Krogan, S. Chung, and et al., "A Bayesian networks approach for predicting protein-protein interactions from genomic data," Science **302,** 449–453 (2003).

[119] D. Rhodes, S. Tomlins, S. Varambally, V. Mahavisno, T. Barrett, S. Kalyana-Sundaram, D. Ghosh, A. Pandey, and A. Chinnaiyan, "Probabilistic model of the human protein-protein interaction network," Nature Biotechnol. **23,** 951–959 (2005).

[120] N. Friedman, D. Geiger, and M. Goldzmidt, "Bayesian Network Classifiers," Machine Learning **29,** 131–163 (1997).

[121] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis* (John Wiley and Sons, New York, 1973).

[122] P. W. Langley and K. Thompson, *Tenth National Conference on Artificial Intelligence* (AAAI Press, Menlo Park, CA, 1992).

[123] N. Friedman, "On bias, variance, 0/1 - loss, and the curse-of-dimensionality," Data Mining and Knowledge Discovery **1,** 55–77 (1997).

[124] X. Lin, M. Liu, and X. Chen, *Proceedings of IEEE International Conference on Bioinformatics and Biomedicine* (2008).

[125] C. K. Chow and C. N. Liu, "Approximating discrete probability distributions with dependence trees," IEEE Trans. on Info. Theory **14,** 462–467 (1968).

[126] T. H. Cormen, *Introduction to Algorithms* (MIT Press, Cambridge, MA, 1990).

[127] M. Remm, C. E. V. Storm, and E. L. L. Sonnhammer, "Automatic clustering of orthologs and in-paralogs from pairwise species comparisons," J. Mol. Biol. **314,** 1041–1052 (2001).

[128] T. Soong, K. O. Wrzeszczynski, and B. Rost, "Physical protein–protein interactions predicted from microarrays," Bioinformatics **24,** 2608–2614 (2008).

[129] J. L. Rodgers and W. A. Nicewander, "Thirteen ways to look at the correlation coefficient," The American Statistician **42,** 59–66 (1988).

[130] F. Provost, T. Fawcett, and R. Kohavi, *Proc. IMLC-98* (1998).

[131] F. Provost and T. Fawcett, *Proc. KDD-97* (1997).

[132] G. Mishra, M. Suresh, K. Kumaran, and et al., "Human Protein Reference Database," Nucleic Acids Research **34,** D411–D414 (2006).

[133] S. Peri, J. D. Navarro, R. Amanchy, and et al., "Development of human protein reference database as an initial platform for approaching systems biology in humans," Genome research **13,** 2363–2371 (2003).

[134] R. Edgar, M. Domrachev, and A. E. Lash, "Gene Expression Omnibus: NCBI gene expression and hybridization array data repository," Nucleic Acids Research **30,** 207–210 (2002).

[135] J. D. Storey, J. M. Akey, and L. Kruglyak, "Multiple locus linkage analysis of genomewide expression in yeast," PLoS Biol **3,** e267 (2005).

[136] G. Yvert, R. B. Brem, J. M. Akey, and et al., "Trans-acting regulatory variation in Saccharomyces cerevisiae and the role of transcription factors," Nat Genet **35,** 57–64 (2003).

[137] R. B. Brem, G. Yvert, R. Clinton, and L. Kruglyak, "Genetic dissection of transcriptional regulation in budding yeast," Science **296,** 752–755 (2002).

[138] L. R. Baugh, A. A. Hill, J. M. Claggett, K. Hill-Harfe, and et al., "The homeodomain protein PAL-1 specifies a lineage-specific regulatory network in the C. elegans embryo," Development **132,** 1843–1854 (2005).

[139] J. J. McElwee, E. Schuster, E. Blance, J. H. Thomas, and et al., "Shared transcriptional signature in Caenorhabditis elegans Dauer larvae and long-lived daf-2 mutants implicates detoxification system in longevity assurance," J. Biol Chem **279,** 44533–44543 (2004).

[140] V. Reinke, H. E. Smith, J. Nance, J. Wang, and et al., "A global profile of germline gene expression in C. elegans," Mol Cell **6,** 605–616 (2000).

[141] R. B. Beckstead, G. Lam, and C. S. Thummel, "The genomic response to 20-hydroxyecdysone at the onset of Drosophila metamorphosis," Genome Biol **6,** R99 (2005).

[142] F. Provost and B. Buchanan, "Inductive Policy: The Pragmatics of Bias Selection.," Machine Learning **20,** 35–61 (1995).

[143] F. Provost, *Proceedings of the AAAI'2000 Workshop on Imbalanced Data Sets* (2000).

[144] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," Journal of the Royal Statistical Society 39 (1977).

[145] G. J. McLachlan and T. Krishnan, *The EM Algorithm and its Extensions* (Wiley, New York, 1997).

[146] G. F. Cooper, *Proceeding of the Fifth International Workshop on Artificial Intelligence and Statistics* (1995).

[147] P. Spirtes, C. Meek, and T. Richardson, *Uncertainty in Artificial Intelligence, Proceedings of the Eleventh Confernce* (San Francisco, California, 1995).

Index