uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Ziying Liu

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.Sc. (Systems Science)

GRADE / DEGREE

Systems Science

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Identifying Gene Regulatory Networks using Bayesian Networks and Domain Knowledge

TITRE DE LA THÈSE / TITLE OF THESIS

Professor Stan Matwin

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Profesor Fazel Famili

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Professor Yongyi Mao

Professor Marcel Turcotte

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Identifying Gene Regulatory Networks Using Bayesian Networks and Domain Knowledge

Ziying Liu

Supervised by:

Dr. Stan Matwin

Dr. Fazel Famili

Thesis

submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the degree of Master of Science in System Science

University of Ottawa
Ottawa, Ontario, Canada
April 2006

Your file Votre référence
ISBN: 978-0-494-18442-4
Our file Notre référence
ISBN: 978-0-494-18442-4

Abstract

Bayesian network techniques have been used for discovering causal relationships among large number of variables in many applications. This thesis demonstrates how Bayesian techniques are used to build gene regulation networks. The contribution of this thesis is to find a novel way of combining pre-knowledge (biological domain information) into Bayesian network learning process for microarray data analysis. Such pre-knowledge includes biological process, cellular component and molecular function information and cell cycle information. Incorporating preexisting knowledge into the Bayesian network learning process significantly improves the accuracy and performance of learning. Another contribution of this thesis is the inference and validation of learning result based on the biological literature and biological knowledge. The learned network structure is presented graphically to make the results easy to understand. A yeast microarray dataset is used to test the performance of the learning process.

Acknowledge

I would like to express my gratitude to my two thesis supervisors Dr. Stan Matwin and Dr. Fazel Famili. Without their assistance, this thesis would not have been possible. I thank them for their patience and encouragement that carried me on through difficult times, and for their insights and suggestions that helped to shape my research skills. Their valuable feedback contributed greatly to this dissertation.

I would like to thank Dr. Roy Walker for his valuable advice at early stage of this research. I would like to thank Dr. Qingyan Liu and Dr. Youlian Pan; they are the ones that I always count on to discuss the detail of a problem, especially, on biological aspects. Their valuable suggestion helped me to improve the dissertation. I acknowledge the help and support of Dr. Chunsheng Yang for his valuable feedback of early version of this dissertation.

I would like to acknowledge the National Research Council for the financial assistance for my research and education.

I would like to thank all the IR group members, especially Biomine team members for the help in many ways.

I would like to acknowledge the members of my committee:

Dr. Fazel Famili (thesis supervisor), National Research Council

Dr. Marcel Turcotte, University of Ottawa

Dr. Stan Matwin (thesis supervisor), University of Ottawa

Dr. Yongyi Mao, University of Ottawa

Their constructive comments and suggestions contributed greatly to this dissertation.

I thank my parents for encouragement and supporting me through all these years.

I thank to my husband and my two daughters for their infinite encouragement, moral support, patience and understanding.

Table of Contents

Chapter 1 Introduction

1.1 Motivation to study gene regulation networks

Biological systems are dynamic organizations, which are controlled by gene regulatory networks. These networks are mediated by the interactions among various biochemical molecules in the cell (e.g. DNA, RNA, protein). During the development and life of an organism, the cell at any given time can respond to the environment and intracellular signals. All cells in an organism have the same genomic data, but the proteins synthesized in each vary according to cell type, time and environmental signals (see Appendix A). In this thesis, the word "cell" denotes a eukaryotic cell.

Current knowledge on molecular interaction networks includes metabolic pathways, regulatory pathways and protein signaling pathways. Understanding the behavior of these networks is important for the discovery of mechanisms underlying the disease process and thus leads to the identification of potential therapeutic targets in drug development and medical treatments. For example, if we know that gene A induces gene B under condition X, then we may prevent it from happening by blocking A, inhibiting gene B or changing the condition X.

The advances in microarray technology have been the driving force for studying genome wide transcript expression using cDNA (complementary DNA) or oligonucleotide microarrays [Velculescu et al 1995, Lockhart et al 1996, Eisen et al 1998, Segal et al 2003]. These advances have allowed researchers to analyze thousands of genes simultaneously at the transcription level instead of one single gene or pathway at a time - the traditional low throughput approach, such as Northern blot analysis. DNA microarray analysis of gene expression of the entire transcriptome at several different time points provides a global view of gene interactions during development or cell cycle progression. Extracting relevant information from these large datasets requires intensive computation and evaluation. Data mining and machine learning techniques are therefore necessary to facilitate knowledge discovery. The objective of this thesis is to apply these techniques for the discovery of the novel gene regulatory networks using time series gene expression data.

1.2 Existing methods in microarray analysis for the regulation networks

Clustering algorithms have been widely used to discover co-regulated genes in microarray data. Clustering analysis is a data mining technique, which attempts to locate groups of objects that have similar pattern over a set of experiments [Everitt, 1974 and Jain et al, 1998]. In gene expression study, this analysis is used to identify meaningful subgroups of genes that behave similarly during a time course or under different treatment conditions. Genes in the same cluster, which have similar expression patterns, may share the common molecular control process or related regulatory pathways. Therefore, if many genes with known functions in a cluster respond to certain experimental conditions (disease or treatment), one could infer that other genes with unknown functions in the same cluster may also be related to the same regulatory pathways. This will provide new knowledge to researchers for the functional studies with the unknown genes.

Several clustering approaches have been applied to microarray data analysis. *Hierarchical clustering techniques* [Eisen et al., 1998 and Ward 1963], start with each single gene as one cluster, then the most similar clusters are merged recursively. It finally ends with one cluster, and yields a tree (dendrogram), which represents nested clusters and similarity levels. These trees can then be 'cut' at different levels to generate disjoint groupings of data. The method of similarity measurement and the order of genes in the initial dataset may affect the final results.

Partitioning optimization techniques (K-means) [Tavazoie et al., 1999 and Famili et al., 2004] K-means partition optimization [McQueen 1967] maintains k cluster centroids, which are summary descriptions of objects in the same cluster. Data objects are assigned to the nearest cluster and the cluster centroids are recomputed iteratively until an end condition is satisfied with a minimized distance within each individual cluster and maximized distances between clusters, under the condition that: no re-assignment of objects, minimal decrease in squared error or an iteration limit is reached. The main limitation of this algorithm is that the number of clusters should be decided in advance.

Principal Components Analysis (PCA) [Raychaudhuri et al., 2000 and Rifkin et al, 2002] is a statistical technique for determining the key variables in a multidimensional dataset that explains differences in the observations made. This approach can be used to simplify the analysis and visualize multidimensional datasets [Pearson 1974 and Everitt et al., 1992].

Self Organizing Maps (SOM) [Kohonen 1989 and Tamayo et al, 1999] uses neural networks to map data objects into a one or two-dimensional lattice in which neighboring nodes tend to define related clusters.

Model based approaches are also used in the analysis of time-series microarray data. These methods consider the dependencies between expression profiles belonging to subsequent time-points. Schliep et al [2003] used an iterative procedure based on HMM (Hidden Markov Model) to find cluster models and an assignment of data points to the models that maximize the joint likelihood of clustering and models. Bar-Joseph et al [2002] proposed an approach based on statistical models: each cluster represented by a spline curve and clustering is computed using an EM-type algorithm (EM: Expectation Maximization). Similarly, Kundaje et al [2002] used a clustering algorithm based on statistical splines to estimate continuous probabilistic models for clusters of genes with similar time series expression profiles, and individual genes. Ramoni et al [2002 a, 2002 b] used a model based clustering approach, in which cluster models were autoregressive curves of a fixed order. Luan et al [2003] introduced a mixed-effects model in analyzing time course gene expression data for performing clustering of genes in a mix model framework. Michaels et al [1998] took advantage of cluster analysis and graphical visualization methods to reveal correlated patterns of gene expression from time series data. Other clustering methods related to this research are graph theoretic techniques [Hartuv et al., 2000] and cluster identification via connectivity kernels [Sharan et al., 2000].

Despite various clustering methods described in the literature, clustering cannot reveal interactions between genes or group of genes; it can only answer questions such as "which genes maybe functionally related?" or "which specific genes are related to the class of sample?". Clustering can also identify a new sample on the basis of its gene

expression pattern, etc. To extract meaningful information from the expression data and discover the interactions between genes or groups of genes, many researches in computational biology have focused on inferring regulatory interactions between genes from time series gene expression data. Finding the relations between genes or groups of genes from huge amount of data is scientifically important and technically challenging. Nevertheless, studies exploring gene regulation networks have been performed with certain degree of success. Some of the frequently used approaches are:

Linear Model: [D'haeseleer et al., 1999 and Someren et al., 2000] assumes that expression level of a node in a network depends on linear combination of the expression levels of its neighbors.

Boolean networks [Liang et al., 1998, Akutsu et al., 1999] assume only two distinct levels of expression, 0 and 1. According to this model, the value of a node at a given step is a Boolean function of the level. A Boolean network consists of n nodes (e.g. representing genes), which either can be repressed or expressed (the node has state 0 or 1, respectively). The dynamics of the network are determined by a list of n (Boolean) functions, each of which receives input from k specified nodes (k varies for each node). Each node has its own specific function, which can determine its next state from the current states of all the input nodes. Compared to gene networks where a gene cannot be in the states of either on or off, Boolean networks are relatively coarse simplifications.

Differential equations: Wahde and Hertz [2000] followed Reinitz and Sharp [1995] in modeling genetic networks as a set of nonlinear differential equations. Thus, they can search for parameters that indicate the rates of change of a certain gene, and the assumption of discrete time steps for the network's next state is unnecessary; instead, they used a GA (Genetic Algorithm) to determine the time constants for each of the n nodes in the system.

Bayesian networks have been developed within the scope of artificial intelligence to construct expert system. Pearl [1988] showed that with Bayesian networks, one could construct networks, which are able to give advice, by reasoning with probabilities, on the basis of observation from the real world. The basic idea is to display the conditional

dependencies and independencies among the variables. The statistical framework of Bayesian learning is suitable for domains with a large number of variables and able to combine prior knowledge extracted from the data. It is also robust for handling noisy data, since it deals with uncertainty.

Friedman et al [2000] introduced the Bayesian networks approach to build gene regulation networks for microarray data. The challenge in microarray data analysis is that the number of experiments (samples) is usually small due to the high cost of microarray experiments and the difficulty of acquiring diseased samples at a given time period [Walker et al., 2003], while the dimensionality, the number of genes whose expression is measured, is very large, usually several thousand. The small sample size cannot provide enough information to build proper network.

Many researches have been studying small sample size and large search space problem. Friedman and his colleagues [Friedman et al., 2000] suggested "Sparse algorithm", in which, they identify a relatively small number of candidate parents for each gene based on simple local statistics, and then restrict the search to networks in which only the candidate parents of a variable can be its parents, resulting in a much smaller search space. The parent candidate list is computed based on the data itself, without domain knowledge applied. People also try to apply domain knowledge to reduce the search space. For example in Ott et al [Ott et al., 2004], the authors limited the number of parents to reduce search space according to the assumption that in biological reality, while the number of children of a regulatory gene may be very high, the number of parents can be assumed to be limited. Hartemink et al [2002] proposed combining location and expression data for gene regulatory network building. Basically, the sequence location information of transcription factor binding sites on the chromosome was considered as constraint for the parent candidate lists during the Bayesian network learning process.

The main objective in this thesis is to build a Bayesian network framework and find a novel way to reduce the search space by using the pre-knowledge extracted from the biological domain.

1.3 Contribution and organization of this thesis

This thesis will demonstrate how the Bayesian techniques are used to build gene regulation networks. The contribution of this thesis is to find a novel way of combining pre-knowledge (biological domain information) into Bayesian network learning algorithm for microarray data analysis. Such pre-knowledge includes biological process, cellular component and molecular function information and cell cycle information. The genes, which are involved in the same biological process and also in the nearby sub-cellular locations, are more likely to interact with one another, and then such genes could have causal relationship. Functionally related genes can also be considered as domain knowledge for building regulation pathways. Incorporating preexisting knowledge into the Bayesian network learning process should significantly improve the accuracy and performance of learning. Another contribution of this thesis is the biological literature validation of the learning results.

Furthermore, this thesis implements a Bayesian network learning algorithm with the consideration of associated pre-knowledge and data preprocessing. The learned network structure is presented graphically to make the results easy to understand. A yeast microarray dataset is used to test the performance of the learning process. One of the goals of this thesis is to integrate Bayesian network learning algorithm into a Bio-Intelligence Project (http://iit-iti.nrc-cnrc.gc.ca/projects-projets/biointelligence_e.html) and help biologists identify causal relationships between groups of genes, and individual genes. This thesis will also provide a foundation to bridge the gap between microarray and clinical data analyses for discovering gene-disease relationships. It could therefore be considered as a contribution to personalized medicine.

The thesis is organized as follows. Chapter 2 gives the biological background of DNA microarray technology and related applications, and the need for the identification of gene regulation networks. Chapter 3 describes the basic algorithm of Bayesian network for learning relationship of variables from a sample dataset, as well as useful approaches to improve the learning performance. Chapter 4 demonstrates the results of applying Bayesian network learning to a synthetic dataset and a yeast dataset. The discussion of the results is also given in this chapter, with special emphasis on the effects of various

learning parameters to learning performance and possible improvement by applying some pre-knowledge to the learning process. Chapter 5 gives the conclusion of the thesis and outlines our future work of using improved Bayesian technique and applying it to other microarray datasets.

Chapter 2 Microarray Technology and Its Applications

Thousands of genes and their products (i.e., RNA and proteins) in a given living organism function in a complicated way that creates the mystery of life. Traditional methods in molecular biology generally work on a "one gene in one experiment" basis, examining and collecting data on a single gene, a single protein or a single reaction at a time. The throughput is very limited and the "whole picture" of gene function is hard to obtain. In the past several years, a new technology, called DNA microarray has been developed by Schena et al [1995] for quantitative monitoring of gene expression patterns. This technology has attracted tremendous interests among biologists because it promises to monitor the whole genome on a single chip so that researchers can have a better picture of the interactions among thousands of genes simultaneously.

2.1 Basic concepts

In order to understand the purpose of a microarray experiment and the nature of the resulting data, it is necessary to first understand some basic concepts of cell and molecular biology. A *cell* (Figure 2.1) is a basic unit of a living organism. The number of cells in an organism varies greatly. For example, a bacterium consists of only one cell, while a human body is an aggregate of more than 10^{14} cells. Cells are classified into prokaryotic cells that lack a membrane-enclosed nucleus (e.g. bacteria), and eukaryotic cells (e.g. human cell) that have membrane-bound nuclei and organelles. A human body contains several types of cells such as red blood cells, white blood cells, epithelial cells, and neural cells etc. The process where a cell undergoes a change to an overtly specialized cell type is called "differentiation". Human cells differentiate at the early stages of development from stem cells. Stem cells retain the ability to divide and differentiate into other cell types. After differentiation, each cell type has a well-defined function. [Weaver, 1999].

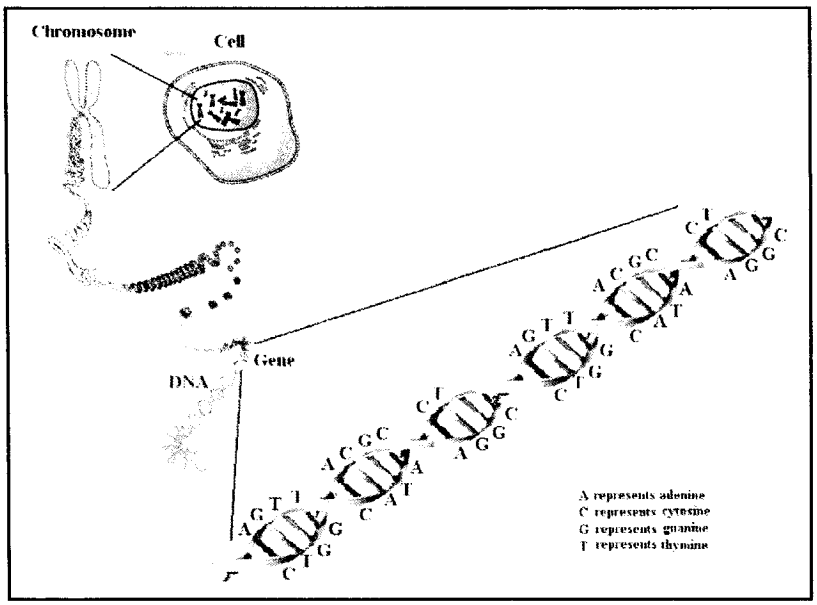


Figure 2.1 The cell, chromosome, and DNA. (*Image credit: National Human Genome Research institute. modified*)

A cell contains several sophisticated organelles that are involved in various cellular processes. These organelles are controlled by an elegant regulatory system whose instructions are coded into deoxyribonucleic acid, or DNA, which is organized in 46 chromosomes contained in the nucleus of each human cell.

An *organelle* is a discrete structure of a cell having specialized functions, such as nucleolus, nucleus, ribosome, and mitochondrion.

The *DNA* (Figure 2.1 and Figure 2.2) serves as storage for hereditary information and it carries this information from generation to generation. Human DNA consists of nucleotides, which are referred to as A (Adenine), T (Thymine), C (Cytosine) and G (Guanine). Nucleotides bind pair-wise (Adenine-Thymine and Cytosine-Guanine) by hydrogen bonds. A pair of nucleotides is referred to as a base-pair (bp). A normal human cell consists of over $3.2 * 10^9$ bps. The complete DNA sequence of an organism is called the genome of that organism.

Gene (Figure 2.2) is the functional and physical unit of the heredity passed from parents to offspring. It is a proportion of DNA sequence, which codes for a functional RNA then protein. Each human gene is made up of bases, A, C, G, and T according to the chemicals

they represent. The number and the order of the bases, determine what we are, how we look and the disease to which we maybe predisposed.

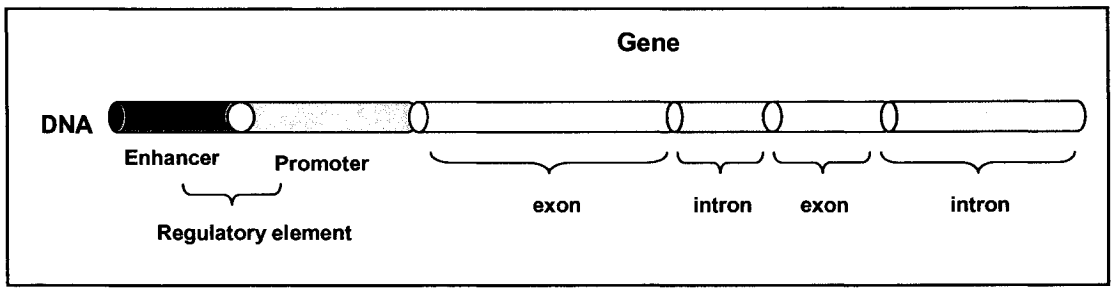


Figure 2.2 The eukaryotic gene has a number of regions, each important to transcription. A specific promoter region upstream of the transcription starting site controls basal level of transcription. A region called enhancer comprised of non-coding DNA increases transcription of RNA initiated by RNA polymerase.

Structure of a gene (Figure 2.2): A gene decomposes into a coding region and a regulatory region. The coding region called exons is the DNA-sequence that encodes a protein, nevertheless, besides coding fragments, called introns, which have no coding information. The introns are sometime called "junk DNA". The function of this "junk DNA" is not yet known. The regulatory region contains binding sites for transcription factors, which affect the process of transcription. As shown in Figure 2.2, the regulatory region can be subdivided into several parts. The promoter determines the sites of transcription initiation and directly bound by RNA polymerase. Promoter proximal elements are control elements close to the start sites. They help regulating transcription of a particular gene. Enhancers have the same function as promoter proximal regions, but they are found at a greater distance from the transcription site, either upstream or downstream of the gene or within an intron. When transcription factors bind to them, the rate of transcription of the gene will be increased. Silencers are control regions of DNA that, like enhancers, may be located thousands of base pairs away from the gene they control. However, when transcription factors bind to them, the rate of transcription of the gene will be decreased. Promoter proximal regions and enhancers are also called cis-regulatory elements. The positive acting proteins, called activator, bind to DNA at or near a promoter site and increase the efficacy with which the RNA polymerase binds to the promoter.

Gene Expression (Figure 2.3 and Figure 2.4): In general, every cell in an organism contains the entire genome, which is subdivided into a set of chromosomes. Each chromosome is a very long continuous piece of DNA that is functionally divided into information units called genes. Each gene carries information for the production of a set of proteins that perform a specialized function in the cell. Proteins are the functional building blocks of an organism.

To retrieve the information encoded in the gene, cells use the process of gene expression, which consists of two steps: in the step called transcription, the gene is copied into the form of mRNA (called messenger RNA), a molecule very similar to DNA. A major player in transcription is an enzyme called RNA-polymerase II (PolII), which creates a new single-stranded mRNA complementary of the DNA sequence. A gene is now represented by a mRNA molecule that is used as a template for the next step. In the step called translation, the mRNA, after being carried out of the nucleus into the cytoplasm, is translated into the corresponding protein. If a gene's corresponding proteins are present in the cell we say this gene is expressed in a cell. During expression many mRNAs are transcribed and translated. The number of mRNAs in the cytoplasm relates to the expression level for that gene and can be used as a measure for the amount of proteins needed by the cell. The mRNA molecules and proteins synthesized during the development and maintenance of an organism, as well as those synthesized in response to the environment are responsible for an organism's characteristics. A gene's up or down regulation means the expression level of this gene in the experimental cells is higher or lower than that in reference cell (control cell or normal cell). Genetic disease is often caused by genes, which are not correctly transcribed: either too much or too little, or which are missing altogether. Such defects are especially common in cancers, which can occur when regulatory genes are deleted, inactivated, or become more active. Microarrays provide a measure of gene activities at the transcription level.

EST (expressed sequence tag) is a fragment of transcript that may not contain an entire coding region. The function of EST is usually unknown.

Transcriptional regulation: A fundamental question regarding gene expression is by which factors it is controlled. For most genes, regulation of transcription initiation is the

principle mechanism for controlling their expression, however gene expression can be controlled at other stages, e.g. by post-transcriptional factors. This thesis focuses on the regulation of transcription.

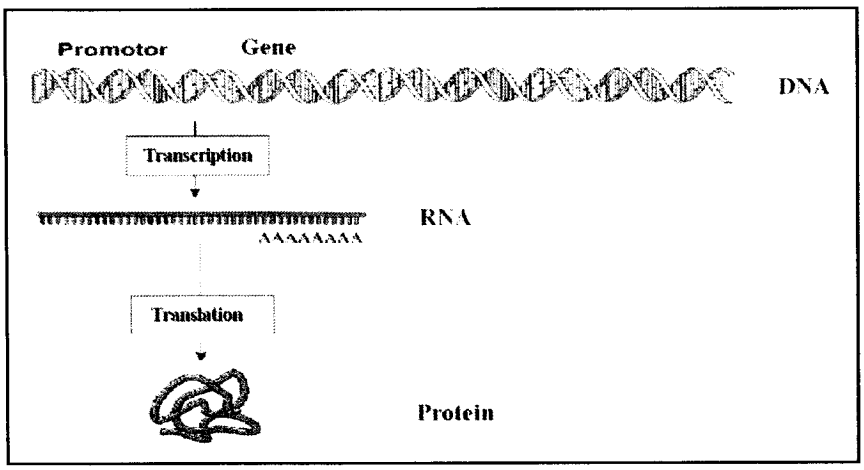


Figure 2.3 Gene expression always involves transcription, which makes RNA. The next step is called translation, which makes protein.

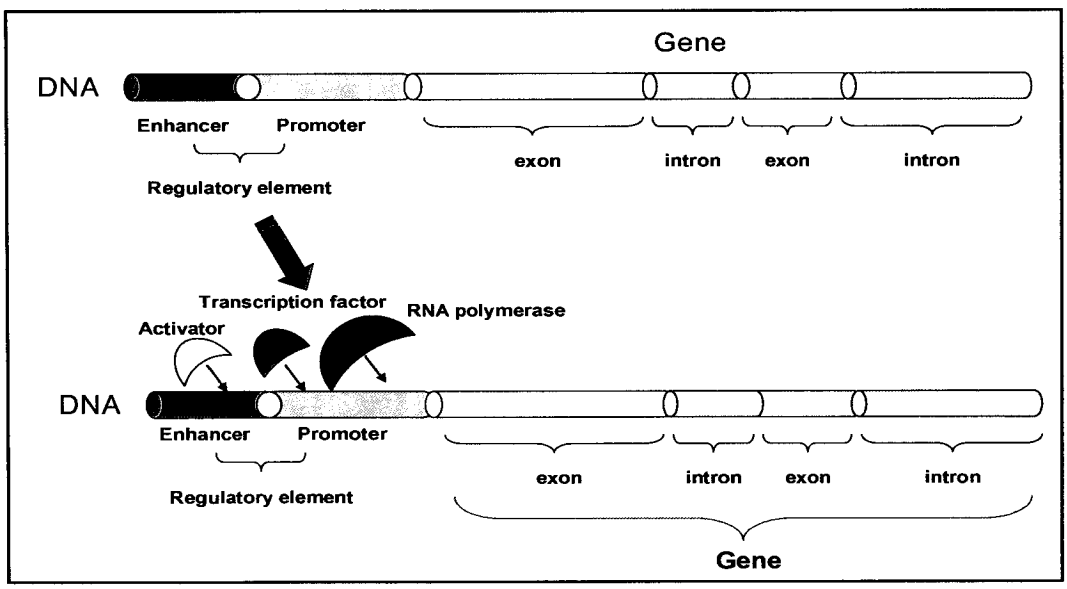


Figure 2.4 RNA polymerase binds to promoter to initiate transcription. Transcription factor binds to the regulatory element or RNA polymerase to facilitate transcription. The binding of activators to the enhancer (or silencer region of the gene if we are going to repress transcription instead) results in bending the DNA molecule so that the activator proteins are brought more closely to the promoter region. This serves to attract more transcription factors to form a transcription complex, thus promote more efficient transcription.

2.2 Microarray technology (hybridization techniques)

The cDNA microarray technology was first introduced by Schena et al [1995]. In this technique, double stranded cDNAs are spotted on a glass slide by a robotic arrayer. The fundamental idea is to compare gene expression levels between two samples, like a tumor and a normal tissue.

In brief, a cDNA microarray experiment proceeds as follows (Figure 2.5).

1. PCR (polymerase chain reaction) amplified selected known genes or ESTs are printed on a glass slide.

2. The entire mRNA population of the test (treated) and reference (control) cells are extracted (mRNA from two kinds of cells: test cell, which comes from disease tissue such as tumor tissue or healthy tissue under certain experiment condition, and reference cell, which comes from healthy tissue). Then the mRNAs are reverse-transcribed to a cDNA and labeled with two fluorescent dyes that absorb clearly distinct wavelengths. It is common practice that cDNA transcribed from the mRNA extracted from treated and control cells are labeled with Cyanine3 (Cy3 - usually red) and Cyanine5 (Cy5 - usually green), respectively. The labeled cDNA molecules then are mixed together and hybridized onto a microarray slide. During hybridization, labeled cDNA molecules bind to their complementary sequences on a microarray slide.

3. After hybridization, the microarray slide is scanned. For each spot, Cy3 and Cy5 intensity levels are measured. The expression rate (Cy3/Cy5) of each gene is measured by scanning the intensity of the color cDNA hybridized to the array. Locations of the cDNA on microarray, or spots, are identified with image analysis software. The resulting image is stored as a 16-bit tagged image file format (TIFF). A red spot indicates an up-regulation of a specific gene in test sample, while a green spot indicates down-regulation of a gene in the test sample. The genes that show equal expression in the test and the reference sample are illustrated as yellow color.

The outcome of a microarray experiment is typically a table of ratios that tells the expression levels of each gene on a microarray between the test (e.g. disease, treatment) and the reference (e.g. healthy, non-treatment) sample. The detail of image analysis and

data preprocessing step to transform the microarray data into normalized data, which represent the expression levels of the individual genes, is beyond the scope of this thesis.

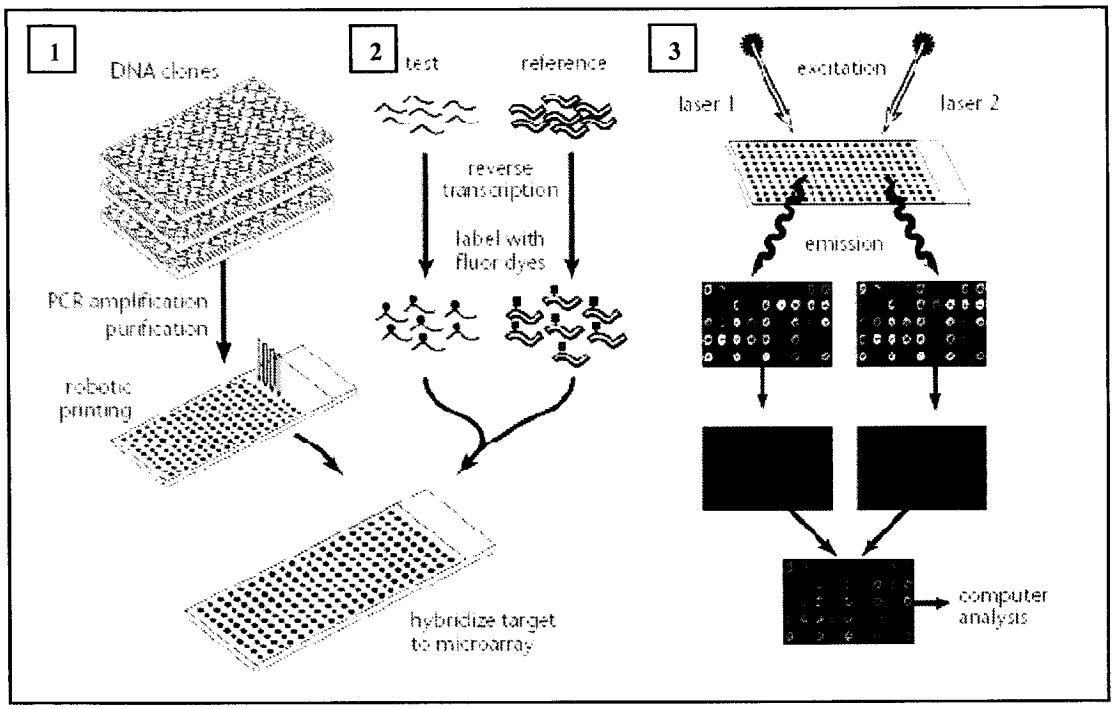


Figure 2.5 Outline of the microarray experiment procedure. [Duggan et al., 1999]

A central concept in microarray studies is "over-expression" or "up-regulation" and "under-expression" or "down-regulation" of a gene. Over-expression of a gene is measured by comparing two samples, for example, cancer and healthy samples, treatment and control, etc. If the amount of mRNA molecules in the cancer sample is larger than in the reference (healthy) sample, the gene is over-expressed. In a similar way, the term "under-expression" or "down-regulation" denotes a situation when a gene produces smaller number of mRNA in a cancer cell than in a healthy cell.

In several diseases, expression levels of hundreds of genes are altered in many cells. Genes performing similar tasks in a cell are related to each other by complex interaction networks, which are referred to as pathways. For example, an interaction network of genes involved in apoptosis is referred to as an apoptosis pathway. A pathway may contain thousands of genes, whose complete set of interactions remain largely unknown. In order to study aberrations in pathways, there is a need to measure expression levels of

thousands of genes simultaneously; such measurements are possible to carry out with microarray technology. Thus, in a study utilizing microarrays, it is possible to research interactions between genes and thereby gain information on cancers and disease treatment at the molecular level. It is a comparison of gene expression difference under two different experimental conditions (treated vs. untreated samples, diseased vs. normal tissues, mutant vs. wild-type organisms, etc). Similarly, the gene expression at different time point could be measured.

2.3 Application of microarray data

The rapid development in microarray technology has launched a revolution in biology. Currently, genomic microarrays are wildly used in bio-medical research [Alam et al., 2001, Cojocaru et al., 2001, Kaminski et al., 2002, Kraft et al., 2003 and Tzouvelekis et al., 2004].

Microarrays are used for biological research, such as gene regulation network and gene function prediction. This technology allows researchers to monitor hundreds, even thousands of genes on a single chip giving scientists a better picture of simultaneous interactions among these genes. Therefore, microarray data analysis can help to determine transcriptional programs of cells for a given cellular function (e.g., cell function, cell differentiation, etc.), molecular pathway, help scientists gain the insight of the activities of cells, and further understand the mechanism of disease development at molecular level.

Microarrays can help researchers to identify the new therapeutic targets for certain diseases by evaluating the changes in the expression levels of many genes simultaneously and to identify distinctive expression pattern characteristic of physiological and pathological states, and to screen for very subtle changes in response to a particular therapeutic treatment.

Microarrays can be used to predict drug side effects (drug toxicity). Microarray data analysis also helps people to understand the correlations between therapeutic responses to drugs and the gene expression profiles of patients, and understand why some drugs work

better in some patients than in others and why some drugs may even be highly toxic (side effect) for certain patients.

Recently, microarrays have been used for medical research. People classify certain tumor or disease by comparing and contrasting gene expression patterns to help diagnosis of diseases or sub-classify certain tumor or disease [Golub et al., 1999, Nutt et al., 2003, Liang et al., 2005]. Microarray data analysis can predict therapeutic response and improve treatments. For example, more accurate classification of a disease could eventually help doctors to provide an accurate way for diagnosis and treatment of diseases, reduce the side effects of conventional drugs and improve patients' survival rate.

Overall, creating public Microarray databases will help us extract the functional information of complex biological systems and facilitate our bio-medical research.

In the future, doctors can just send a patient's blood or a tissue sample to the lab, the result of the microarray experiment will assist doctors to precisely diagnose the disease based on the gene expression, and provide a more precise way to treat diseases, and get rid of potentially debilitating side effects of conventional drugs. For example, people study the classification and relation of genes based on measurements of gene expression under certain condition. A potential of this work is in the analysis of cancer patient data. It is hoped that more accurate classification of tumors will result in improved therapeutic design including decisions on when additional treatment may be necessary and effective.

2.4 Challenges of microarray data

A microarray dataset may consists of millions of data points, which are usually noisy and may be corrupted by errors from several sources [Troyanskaya et al., 2001], such as (i) quality of microarray itself (robotic methods used to print the microarray), (ii) the method used in the experiment (e.g. insufficient resolution during the experiment), (iii) labeling reaction problem, (iv) scanning and (v) measurement problem (e.g. machine or operator), or (vi) due to dust or scratches on the slide as well as other unknown experimental factors. In addition, microarray data contain much more variable than sample size. This is known as a "fat array" and is a common challenge in Machine Learning from gene

expression data. In Machine Learning, we know that the size of the search space increases exponentially with the number of parameters of the model. Therefore the more variables used in a model, the harder the modeling task becomes. This is also a big challenge in Bayesian network learning, which we will further discuss in the following chapters.

Chapter 3 Methodologies

This chapter presents the principle of Bayesian network technique, including score function and search methods. Other techniques involved in Bayesian network learning such as data preprocessing, missing data handling as well as some implementation issues are also covered.

3.1 Introduction on Bayesian networks

A Bayesian network is an acyclic graphical model that encodes the relationship among a set of variables. A simple example of such graphical model is shown in Figure 3.1, which has seven nodes connected by six directed edges.

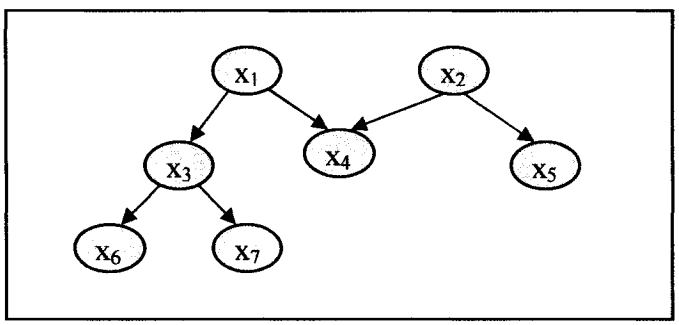


Figure 3.1 Example of a simple Bayesian network

Each node in this model represents a variable denoted as x_i and a directed edge connecting two nodes represents the conditional dependency of a child variable with respect to a parent variable. There are two sets of information provided in a Bayesian network:

- *Conditional dependency or independency.* The conditional dependency of two nodes is represented by the directed edge connecting the two nodes. If all parents of a specific node are known, the node is independent of all other nodes that are not its descendant. If two nodes are not connected by a directed edge, then they are conditionally independent.

- *Conditional probabilities of a child node versus each of its parents.* For example, $P(X_3|X_1)$ represents the probability of x_3 having a value X_3 given x_1 having a value X_1 (i.e. X_i represents the instantiation of x_i).

An example of Bayesian network is given in Figure 3.2. This Bayesian network, which describes the dependence relationship among eight variables, has been used by researchers to test and evaluate different algorithms involved in Bayesian network learning.

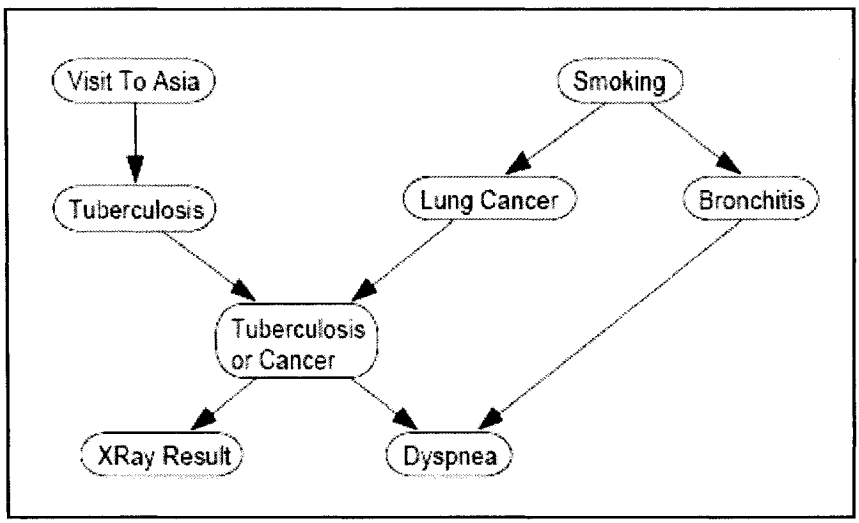


Figure 3.2 *Chest Clinic* network adopted from http://www.norsys.com/, proposed by Spiegelhalter [1986].

Original conditional probability for the above network is given in Table 3.1, which shows the conditional probabilities of each node with respect to various instantiations of its parent node(s). For example, the variable "Smoking" is the parent of variable "Lung Cancer", indicating "Smoking" has direct effect on "Lung Cancer". When "Smoking" is TRUE, the chance of getting "Lung Cancer" is 10%, while "Smoking" is FALSE, the chance is down to 1%.

P(VisitAsia = TRUE) = 0.01;
P(VisitAsia = FALSE) = 0.99

P(Smoking = TRUE) = 0.5;
P(Smoking = FALSE) = 0.5

P(Tuberculosis = PRESENT | VisitAsia = TRUE) = 0.05;
P(Tuberculosis = ABSENT |VisitAsia = TRUE) = 0.95
P(Tuberculosis = PRESENT |VisitAsia = FALSE) = 0.01;
P(Tuberculosis = ABSENT |VisitAsia = FALSE) = 0.99

P(Lung Cancer = PRESENT | Smoking = TRUE) = 0.1;
P(Lung Cancer = ABSENT | Smoking = TRUE) = 0.9
P(Lung Cancer = PRESENT | Smoking = FALSE) = 0.01;
P(Lung Cancer = ABSENT | Smoking = FALSE) = 0.99

P(TbOrCa = TRUE | Tuberculosis = PRESENT, Lung Cancer = PRESENT) = 1.0
P(TbOrCa = FALSE | Tuberculosis = PRESENT, Lung Cancer = PRESENT) = 0.0
P(TbOrCa = TRUE | Tuberculosis = PRESENT, Lung Cancer = ABSENT) = 1.0
P(TbOrCa = FALSE | Tuberculosis = PRESENT, Lung Cancer = ABSENT) = 0.0
P(TbOrCa = TRUE | Tuberculosis = ABSENT, Lung Cancer = PRESENT) = 1.0
P(TbOrCa = FALSE | Tuberculosis = ABSENT, Lung Cancer = PRESENT) = 0.0
P(TbOrCa = TRUE | Tuberculosis = ABSENT, Lung Cancer = ABSENT) = 0.0
P(TbOrCa = FALSE | Tuberculosis = ABSENT, Lung Cancer = ABSENT) = 1.0

P(XRay = ABNORMAL | TbOrCa = TRUE) = 0.98;
P(XRay = NORMAL | TbOrCa = TRUE) = 0.02
P(XRay = ABNORMAL | TbOrCa = FALSE) = 0.05;
P(XRay = NORMAL | TbOrCa = FALSE) = 0.95

P(Bronchitis = PRESENT | Smoking = TRUE) = 0.6;
P(Bronchitis = ABSENT | Smoking = TRUE) = 0.4
P(Bronchitis = PRESENT | Smoking = FALSE) = 0.3;
P(Bronchitis = ABSENT | Smoking = FALSE) = 0.7

P(Dyspnea = PRESENT | TbOrCa = TRUE, Bronchitis = PRESENT) = 0.9
P(Dyspnea = ABSENT | TbOrCa = TRUE, Bronchitis = PRESENT) = 0.1
P(Dyspnea = PRESENT | TbOrCa = TRUE, Bronchitis = ABSENT) = 0.7
P(Dyspnea = ABSENT | TbOrCa = TRUE, Bronchitis = ABSENT) = 0.3
P(Dyspnea = PRESENT | TbOrCa = FALSE, Bronchitis = PRESENT) = 0.8
P(Dyspnea = ABSENT | TbOrCa = FALSE, Bronchitis = PRESENT) = 0.2
P(Dyspnea = PRESENT | TbOrCa = FALSE, Bronchitis = ABSENT) = 0.1
P(Dyspnea = ABSENT | TbOrCa = FALSE, Bronchitis = ABSENT) = 0.9

Table 3.1 Conditional probabilities of *Chest Clinic* network

The conditional independence or dependence among variables encoded in a Bayesian network provides a simple way to determine the probability distribution of the variables in the network. The joint probability distribution of all n variables in a Bayesian network is given by the following formula:

$$P(X_1, X_2 \ldots X_n) = \prod_{i=1}^{n} P(X_i \mid \pi_i) \qquad (3.1)$$

Where X_i represents the instantiation of variable x_i and π_i represents the instantiation of all parents of x_i. Probability $P(X_i \mid \pi_i)$ can be determined from the conditional probabilities associated with each directed edge linking x_i with its parents. The joint probability of any instantiation of all variables in the network can then be computed as the product of n probabilities. For example, in the Bayesian network in Figure 3.2, the joint probability of all eight variables can be expressed as:

P(VisitToAsia, Smoking, Tuberculosis, LungCancer, TbOrCa, Xray, Bronchitis, Dyspnea) =
P(VisitToAsia) * P(Smoking) * P(Tuberculosis | VisitToAsia) * P(LungCancer | Smoking) *
P(TbOrCa | Tuberculosis, LungCancer) * P(Xray | TbOrCa) * P(Bronchitis | Smoking) *
P(Dyspnea | TbOrCa, Bronchitis)

The above expression provides the joint probability of the dataset in a compact and easy to handle form. If a joint probability table were used, those eight boolean variables would need $2^8 = 256$ rows to cover all the combinations. For a dataset with a larger number of variables and each variable having more instantiations, the number of rows in the joint probability table increases exponentially and the table is very hard to establish.

Furthermore, the joint probability function given in (3.1) can be decomposed into the probability of each individual node, which allows simplifying the complexity of computation in building Bayesian network.

Because of their simplicity in describing the relationship of variables in a dataset, Bayesian networks have been used in applications that involve discovering relationships among a large number of variables. Bayesian networks are also considered as causal network because they provide causal effect among variables that is determined by the conditional dependency [Friedman et al., 2000]. In this thesis, we use Bayesian network techniques to establish the relationships among genes in a microarray dataset.

There are two main processes involved in Bayesian network applications: learning and inference. Learning is the process of building a Bayesian network from a sample dataset and/or from some existing knowledge. Inference is the process of inducing probabilities among variables given a Bayesian network, most commonly the conditional probabilities of one variable given a set of some other variables. This thesis is concerned only with the Bayesian network learning process because our goal is to find the causal effect among genes in a given microarray dataset.

3.2 Learning Bayesian networks with score function

Bayesian networks have advantage of representing probability distribution function of a dataset with a simple compact model. However, building such networks is not an easy task. Methods that have been developed for building Bayesian networks can be clustered into two categories. The first approach uses knowledge from domain experts to manually build a network. An accurate network can be built with this approach. However this method needs domain expertise, which is not always available, especially for some new areas. Another problem is that the method is only feasible for datasets with small number of variables.

The second approach is to build Bayesian networks from an existing dataset, usually with a search process by using probability information calculated from the dataset. This approach can be used for building Bayesian networks for dataset of any size, and the learning process may be resource consuming for large dataset. Many research efforts have been made to learn Bayesian networks from existing data or experiment data. This section will describe one such method developed by Cooper and Herskovits [1992], which is to find the structure that maximizes the posteriori probability for a given dataset among the space of possible network structures.

Let B_s represent the Bayesian network (which includes the structure and the conditional probabilities encoded in the structure), and D represents the given dataset, the posterior probability is given as:

$$P(B_s \mid D) = \frac{P(B_s, D)}{p(D)} \tag{3.2}$$

where $P(D)$ is the probability of the values of variables in a given dataset. This probability is considered to be the same for all the network structures examined and is not relevant in calculation of $P(B_s|D)$. So finding maximum posterior probability of a Bayesian network become the search for the maximum probability $P(B_s, D)$. This search contains two parts: (i) finding the best network structure and (ii) finding the best conditional probabilities among parameters for that structure. Given a network structure, the conditional probability is denoted as $P(B_p|B_s)$, where B_p represents the set of parameters of the network structure. To deduce an efficient formula for computing the probabilities $P(B_p|B_s)$ and $P(B_s, D)$, Cooper and Herskovits [1992] have introduced four assumptions:

1. *The variables in the dataset are all discrete. For continuous variables, such as the gene expression level in microarray dataset, a prior discretization step is required.*
2. *Each experiment (e.g. patient, condition) in the dataset is not influenced by any other experiments. A dataset usually contains the data from different experiments. These experiments must be independent.*
3. *The dataset is complete, i.e. there is no missing data in the dataset. (We will present some methods in the following section to address the case where there is missing data in the dataset).*
4. *The probability distribution of $P(B_p|B_s)$ must be uniform.*

Based on the assumptions above, Cooper and Herskovits [1992] have deduced the following formula for the joint probability $P(B_s, D)$:

$$\mathrm{P}(\mathrm{B_s}, \mathrm{D}) = P(B_s)\prod_{i=1}^{n}\prod_{j=1}^{q_i}\frac{(r_i-1)!}{(N_{ij}+r_i-1)!}\prod_{k=1}^{r_i}N_{ijk}! \tag{3.3}$$

where:

n – total number of variables in the dataset

i – the index of the current variable x_i

q_i – the number of parent instantiations π_i of variable x_i (see equation 3.1)

r_i – the number of instantiations of variable x_i

j – the index of current parent instantiation

k – the index of current instantiation of x_i

N_{ij} – the number of cases in the dataset that match with the parent instantiation j

N_{ijk} – the number of cases in the dataset that match with the instantiation k of x_i and its parent instantiation j.

Obviously, the relationship between N_{ij} and N_{ijk} is given by:

$$N_{ij} = \sum_{k=1}^{r_i} N_{ijk} \tag{3.4}$$

The first term in formula (3.3) is the probability of a specific network structure. This probability is usually the same for all possible structures, and is not relevant in the calculation of the posterior probability. The rest of the formula contains only various numbers of variable instantiations. Therefore the calculation of the posterior becomes simply counting the numbers of related instantiations in the dataset. Formula (3.3) is called score function and the Bayesian network learning is to find, among all possible structures, the network structure that maximizes score function.

3.3 Search methods

As presented in the previous section, learning a Bayesian network consists of searching the entire space of possible network structures for the one that maximize score function. However, this search is an NP-hard [Chickering, 1996] task considering the number of the potential structures that the search process needs to look at, especially for datasets that have large number of variables where the number of potential structures increases exponentially.

In Particular, [Robinson, 1977] derived the following efficiently computable recursive function for determining the number of possible belief network structures that combine n nodes.

$$f(n) = \sum_{i=1}^{n} (-1)^{(i+1)} \binom{n}{i} 2^{i(n-i)} f(n-1) \tag{3.5}$$

The number of possible structures grows exponentially as in Table 3.2:

Number of nodes	Number of structures
2	3
3	25
5	29000
10	$4.2*10^{18}$

Table 3.2 Number of possible structures with respect to number of nodes

Thus an exhaustive enumeration of all network structures is not feasible in most domains. One efficient approach that is commonly used in learning Bayesian network is to use heuristic search to attempt finding a network that is, or close to be the best one. Such a search usually starts with a structure that can be empty or contain some edges specified by user according to certain pre-knowledge. Then changes are made to this network structure, and the score function defined in (3.3) for each change is evaluated. The possible changes include adding a new edge, deleting or reversing an existing edge in the current structure. If an improvement is found when making a change, then the change will be included into the current structure and search continues with other changes to be attempted.

Note that this search method is only looking for the neighboring space of the current structure. Each attempted move is considering adding, deleting or reversing only one edge. The advantage of this search method is that it requires less computation of the score function when attempting each change. As mentioned earlier, the calculation of the score of each node consists of collecting the counts of related instantiations in the dataset, which is the most resource consuming part in the entire search process. The score function defined in equation (3.3) is the product of the scores of each individual node, only the nodes that are involved in each change need to recalculate their score, while the score of other nodes remains unchanged. The following sub-sections will present some of the heuristic search methods.

3.3.1 Greedy search

Starting with an initial structure, this method chooses a random change to the current structure, usually by selecting two random nodes and changing the relationship between

the two nodes. If improvement is found with the random change, i.e. the score function increases, the change will be included into the network structure to make a new current structure and the search continues. If no improvement is found, another random change will be selected and tested. A number is defined as the maximum number of random changes. If no improvement can be found after maximum number of tests has been performed, the current structure is then the final Bayesian network. The pseudo code is given in Figure 3.3.

```
Initial structure S
Compute MaxScore = Score(S)
Set i = 0
While (i < MaxChanges)
        New structure S' by random change
        If Score(S') > MaxScore
                Set MaxScore = Score(S')
                Set S = S'
        End If
        Set i = i + 1
End while
```

Figure 3.3 Pseudo-code of greedy search

The advantage of greedy search is the speed of the search process, since it does not require searching all the neighboring space at each step by simply testing a fixed number of the changes to the current structure. Obviously the maximum number of changes should increase properly along with the number of variables in a dataset. The problem with the approach is that it can be easily stuck in local optima, especially when the maximum number of changes is not high enough.

3.3.2 Simulated Annealing (SA) search

Various techniques have been developed to help find global optima in search space. Simulated annealing search is one of the methods that can avoid the problem of getting stuck in local optima [Metropolis et al. 1953, Kirkpatrick et al. 1983 and Cerny 1985]. This method is based on the similarity of the search process to the process of metal cooling into a crystalline structure. SA method starts with an initial network structure and an initial temperature. Same as in the greedy search, a random change is selected and the score function is computed with Δ representing the score change. But unlike greedy

search where the change is accepted only when Δ is positive (score increases), SA may also accept a change with a negative Δ based on a certain probability defined as:

$$p = e^{\frac{\Delta}{T}}$$

So a change is accepted when $p > 1$ (i.e. $\Delta > 0$) or the change is accepted with probability p. In each search step, this evaluation is repeated α times or until β changes have been accepted. If no change has been found during α evaluations, the search process will be stopped. Another stop condition is when the temperature has been lowered δ times. The temperature value T is set to T_0 at the first search step, and decays, at each following search step, with a factor γ ($0 < \gamma < 1$). This means that a change is easy to be accepted at first search step when the temperature is higher, and when the temperature comes down, it becomes harder to accept a negative change.

Compared to greedy search where only one parameter is required, the SA search uses five parameters. The pseudo-code of SA search is given in Figure 3.4.

```
Define α: maxi # of changes;    β: #of accepted changes,
       γ: temperature lowering rate,
       δ: maxi # of lowering temperature and
       T0 : initial temperature,
Initial structure S
Compute MaxScore = Score(S)
Set i = 0
While (i < δ)
       Set j = 0
       Set k = 0
       While (j < α and k <β)
              New structure S' by random change
              Set Δ = Score(S') - MaxScore
              Set p = e^(Δ/T)
              If (p > 1 or p > Random (0, 1))
                     Set MaxScore = Score(S')
                     Set S = S'
                     Set k = k + 1
              End If
              Set j = j + 1
       End While
       Set i = i + 1
       Set T = T * γ
End while
```

Figure 3.4 Pseudo-code of SA search

3.4 Data pre-processing

Bayesian networking learning is a time consuming task, especially when the number of variables are too many (Table 3.2). Making proper pre-processing and including some pre-knowledge into the dataset would improve the learning performance for both computational resource and result accuracy.

3.4.1 Discretization

The Bayesian network learning algorithm that has been presented in previous section is only suitable for discrete data (which is one of the assumptions made by Cooper and Herskovits [1992] to develop the score function). In the real world, many datasets contain continuous data, such as microarray dataset. The continuous data must be discretized into a number of states.

There are many different methods for data discretization. The simplest one is range based discretization, which breaks up a variable into equally sized ranges from minimum to the maximum value, and each range is assigned a specific symbol or value.

3.4.2 Data filtering

Not all of the data in a dataset is useful for learning Bayesian networks. This can happen especially for microarray data when the continuous expression levels are discretized. If the gene expression does not have variation across the time axis, this gene does not provide useful information for the search process, but only increases the search space. These genes should be removed from the dataset before starting the learning process so that the search will be attempting only within eligible variables. Whether or not to remove a variable depends on the discretization method used and the value range that is chosen. A smaller value range often results in filtering out fewer variables.

3.4.3 Pre-ordering

Cooper and Herskovits [Cooper and Herskovits, 1992] developed a search algorithm that requires organizing all variables into a specific order table. For each individual variable, only the variables that precede it in the order table can be its parent(s). Therefore, the search space is largely reduced, and the search process on this pre-ordered dataset becomes the following: the initial starting structure is an empty structure; for each node, all eligible parent candidates determined by the ordered table are tested and the one that provides the best score function will be added to the parent list for that node. The process continues for the same node with the remaining parent candidates until there is no improvement for that node or all parent candidates have become the parents of the current node. The entire search completes when all variables (nodes) have been processed in the pre-defined order. The pseudo-code of pre-order search is given in Figure 3.5.

```
Initial structure S
Compute MaxScore = Score(S)
Set i = 0
While (i < NumberOfNodes)
        Set ParentList = 0
        Set NoImprovement = FALSE
        While ((ParentList = all parent candidates) Or (NoImprovement = FALSE))
                Set j = 0
                Set NoImprovement = TRUE
                While ((j < i) and (ParentList no contain i))
                        New S' by adding i to ParentList
                        If Score(S') > MaxScore
                                Set i to ParentList
                                MaxScore = Score(S')
                                Set NoImprovement = FALSE
                        End If
                        Set j = j + 1
                End While
        End While
        Set i = i + 1
End While
```

Figure 3.5 Pseudo-code of pre-ordered search

The advantage of the pre-ordered dataset is that the search space is much smaller than the regular dataset and the search performance is significantly improved. Note that the learned network cannot be stuck in a local optimum since the search process is looking at the entire space for the best one at each node. The downside of this approach is the difficulty of defining the order of variables, which is critical to the approach. Some simple datasets can have an eligible order for their variables. For example, to use pre-order approach for the *Chest Clinic* network given in Figure 3.2, we should place variable "Smoking" before variable "LungCancer". Since it is clear that if a relationship exists between the two, it must be that: "Smoking" has effect on "LungCancer". For most datasets, however, it is usually hard to define such order, especially for datasets with large number of variables (such as microarray data).

3.4.4 Applying domain knowledge (reduce search space)

In a microarray dataset, while a parent variable can have many children, the number of parents of a given child is very limited. In fact, the eligibility of a variable being a parent of another variable can be partially determined prior to the search process based on pre-

knowledge of the dataset. If we apply this knowledge to limit the number of parents, the number of calculation of the score function can be reduced significantly.

It is shown in [Ott et al, 2004] that the desired Bayesian network can be built using O(n * 2^n) dynamic programming steps, where n is the number of variables in the dataset. However, if we define a constant m ($m \in N$), and a parent candidate lists C, where $|C| \leq m$, then the search process can be complete within $O(2^n)$ steps.

We are going to introduce a parent candidate list in our implementation. Same as for pre-ordered method, determining a parent list is not trivial. Fortunately, for microarray data, there exists some pre-knowledge that can help to build a proper parent candidate list. We will demonstrate in the next chapter how this provides the improvement to the learning performance.

3.5 Missing data handling

The existence of missing data is a common phenomenon in many datasets and may cause a major problem to some data analysis techniques (e.g. k-mean clustering, SOM, hierarchical clustering, PCA). Missing data is due to experiment defects, and in many cases, the exact source of missing data is not known or may be due to a number of causes (see section *2.4 Challenges of microarray data*).

Since the score function that has been presented is only suitable for complete datasets (which is one of the assumptions made by Cooper and Herskovits [1992], to develop the score function), the missing data problem must be addressed before applying the learning algorithm to a dataset that has some missing values. There are a large number of methods that have been developed to handle missing data problem. Basically two approaches can be considered for handling this problem for the Bayesian network learning: imputation and expectation maximization.

3.5.1 Imputation

Many imputation methods have been developed for replacing the missing data with some approximate values. The following is a summary list of the methods that are widely used in various applications:

- Hot deck – The missing data is replaced by the value in another sample where the variable is not missing and which is the closest sample to the sample with a missing value
- Mean substitute – The missing data is replaced by the mean value of the same variable in all other experiments where the variable is not missing
- KNN – K nearest neighbors are selected and the missing variable is replaced by the weighted or non-weighted average of the variable value from the K neighbors
- Regression substitute – The missing variable in a sample is replaced by predicting values from the regression on the known variables for the same sample

Note that imputation is a simple approach to handle missing data in data processing. It is usually used as part of data pre-processing. The problem with imputation is that it can be used only when the missing data are a small portion of the entire data and the information on the missing data can be extracted from the uncompleted data. When the proportion of missing data gets bigger, it is difficult to get enough information by imputation, and some other algorithm must be used.

3.5.2 Expectation Maximization

Expectation maximization is a common approach for dealing with missing data in many applications. As documented in many literatures, this method consists of two steps. First in the expectation step (E), the expected value of the complete data likelihood is computed. In the maximization step (M) the expected values for the missing data obtained from E step is used to replace the missing value and make a complete dataset. Then the new parameters are estimated by maximizing the likelihood based on the completed data. These two steps are repeated until convergence is obtained.

Friedman [Friedman, 1997] has developed a method to allow using EM algorithms in Bayesian network learning. Each search step uses the EM algorithm to estimate the network parameters (conditional probabilities). All the search methods can be used just as with completed data, with the only exception that the score is calculated by using expected sufficient statistics from EM algorithm.

Note that unlike the imputation, the EM algorithm is not looking for substituting the missing data, but only for estimating the parameters (conditional probabilities) at each search step and the parameters will be used in the next search step to find new network structure.

3.7 Implementation issues

So far we have discussed the Bayesian network algorithms and other related techniques that can be used in the learning process. This section addresses the issues of the implementation of the search methods, in particular, the issues that are related to the performance of the learning process.

3.7.1 Input and output format

The implementation provides support for two data formats: Arff (Attribute-Relation File Format) [Witten and Frank, 2000] and Biominer data format.

Arff format is a plain text data format, which is easy to check since it can be loaded into any text editor. The dataset contains two parts: header and data. The header contains the information that describes the data that follows, and most important part is the list of attributes that are contained in the data and all eligible values for each attribute, which make the data loading very simple. The data part is simply row of data separated by space.

Biominer data format is used in Biominer software [Walker et al., 2003]. Similar to Arff data, it is a plain text data format with a three lines header followed by a data part, which is a matrix of m columns and n rows. Supporting Biominer data format is for the eventual integration of Bayesian network techniques into the Bio-Intelligence project, which is one of the goals of this thesis.

The learning process generates a graphic and a text output. The graphic output displays the learned network structure with nodes connected by directed edges. The graph is built using free distributed software named Grappa [Ellson et al., 2003], which provides the layout of the structure based on the learned results. The advantage of the graphic output is that it provides an intuitive view of the learned network. The downside is that it becomes

difficult to check the relationship between one variable and other variables when the number of variables increases, which results in a network graph full of many tangled lines and nodes. However, the text output provides a list of variables, associated with its parent variables. Therefore, is easy to see the parent variables for every individual variable.

3.7.2 Computation of score function

The search process starts by looking for the best score among the neighboring structures that are around the current structure. Computing the score function is the major task that takes most of the computing time. As noted, for each neighboring structure, only the changed variables are involved in the score computation and the score for other variables remains unchanged. The score of a changed node is given as following:

$$S_i = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \qquad (3.5)$$

First, this formula needs a number of computations in the form of a factorial. To save computing time for these factorials, all required factorials are pre-calculated before the search process starts and stored into an array so that they can be retrieved simply by looking into the array.

To calculate the score function of a particular node defined in equation (3.5), the main task consists of collecting all the N_{ijk} for that node, which is the occurrence of each of the node's values under each of its parent instantiations in the entire dataset. There are two ways that can be used to collect this information and calculate the score for the node. The first way is to allocate an array that will be used for containing all N_{ijk}, and to scan each data row in the dataset, and to find the location in the array that matches the values in the row, and to increase the value at that location. The advantage of this approach is that the dataset is scanned only once to determine the entire N_{ijk}. Once we have all the N_{ijk}, the calculation of the formula is straightforward.

Cooper and Herskovits [Cooper and Herskovits, 1992] showed that in case that all N_{ijk} are collected, the time for computing the score is in order of O(m*n*r), where m is the maximum parent instantiations, n and r were defined in equation (3.3). One problem with

collecting all N_{ijk} is that the size of the array required for storing N_{ijk} becomes too big when the number of the parents of each node is bigger. For example, if each variable in a dataset can have three values and each variable can have up to 20 parents, then the number of parent instantiations for any node can be up to $3^{20} \approx 3$GB. This means we need to allocate an array with 3GB element (with each element taking 4 bytes), which is not feasible in today's computers.

An alternative to avoid big memory usage in allocating array for all N_{ijk}s is to recursively scan the dataset, and search for specific N_{ijk}s during the scanning. Since this requires scanning the entire dataset many times to determine all needed N_{ijk}s, this approach is a time consuming process.

3.7.3 Statistics collection

Search methods that are used usually cannot converge to the same optimum each time when running a learning experiment for the same dataset. To ensure the correctness of learning results, each experiment has to be run a number of times. The statistics then will be used to infer the final result based on the confidence level. From the data mining point of view, the confidence level (which is also referred to as accuracy in this thesis) is defined as the number of instances that are correctly learned and expressed as a proportion of all learned instances [Witten and Frank, 2000].

Chapter 4 Results and Discussion

This chapter will give some learning results using the proposed methods for different datasets. First we will use a synthetic data to test and validate the different proposed algorithms and the implementation, in particular, to check the performance of learning under various learning parameters. Then a subset of the well-known yeast dataset [Spellman et al., 1998] will be used to build a Bayesian network to identify the regulation effects among some of the genes in the dataset. We will discuss and validate the learning results using biological knowledge.

4.1 Bayesian network learning from synthetic data

In this section, we are going to test the proposed Bayesian network learning method using a synthetic dataset named *Chest Clinic* mentioned previously (see section *3.1 Introduction on Bayesian networks*). The original network is displayed in Figure 3.2.

This network contains 8 variables (nodes) connected by 8 directed edges representing the causal effect among the variables. For example, variable "Smoking" is a direct cause of variable "Lung Cancer", while "Visit to Asia" has no direct effect on variable "Lung Cancer". The test datasets are generated from the original network based on conditional probability distributions defined for each of the edges. The network of *Chest Clinic* is a typical network that is commonly used by researchers for testing and validating different Bayesian algorithms. Here we apply different learning approaches described in the previous chapter to this dataset and the learning results will be used to compare to the original network to evaluate learning performance.

In order to evaluate the proposed Bayesian network approach, we scored each learned network using the number of edges that are correctly and incorrectly identified and compared these numbers with the original structure. The following numbers were used for the evaluation:

- Number of correct learned edges – the number of edges that appears in both original and learned networks with correct direction

- Number of missing edges – the number of edges that appears only in the original network
- Number of added edges – the number of edges that appears only in the learned network
- Number of reversed edges – the number of edges that appears in both original and learned networks, but the direction is different

Our experiments with the *Chest Clinic* dataset were focused on analyzing the performance of different learning approaches (Greedy, SA and pre-order), the effects of sample size on learning performance and the improvement of learning when applying pre-knowledge. Since the search methods may converge to different optimum in each individual search, we ran the learning process 20 times for each experiment and collected the statistics. We define a confidence level for each edge in the learned network as the number of appearance of an edge among these 20 runs. Obviously only the edges with higher confidence level should be considered acceptable.

4.1.1 Results of different search methods

This experiment uses a *Chest Clinic* dataset that has 1000 samples (generated using the original probability distribution from http://www.norsys.com/). The dataset was applied respectively to pre-ordered search, greedy search and SA search. Table 4.1 and Table 4.2 show the statistics results (with confidence levels) of greedy and SA searches respectively. Note that there is no statistics for pre-ordered approach because there is no random effect in this approach and its result remains the same for a given dataset. In the tables, each row represents a parent variable and the columns in that row are the children variables under that parent associated with a confidence level. For example, variable "Visit To Asia" is the parent of variable "Smoking" with confidence level of 0.65, while "Visit To Asia" cannot be parent of "Lung Cancer" as the confidence level of this relationship is zero.

Children / Parents	VisitToAsia	Smoking	Tuberculosis	LungCancer	TbOrCa	XRay	Bronchitis	Dyspnea
VisitToAsia	0	0.65	0	0	0	0	0.5	0.4
Smoking	0	0	0	0.1	0.15	0	0.55	0.35
Tuberculosis	0	0.3	0	0.25	0.7	0.35	0.4	0.45
LungCancer	0	0.6	0.3	0	0.55	0.35	0.15	0.3
TbOrCa	0	0.35	0.3	0.25	0	0.65	0.2	0.5
XRay	0	0	0	0	0.1	0	0	0.2
Bronchitis	0	0.3	0	0.1	0	0	0	0.5
Dyspnea	0	0	0	0.1	0	0	0.45	0

Table 4.1 Learning statistics of Greedy search

Children / Parents	VisitToAsia	Smoking	Tuberculosis	LungCancer	TbOrCa	XRay	Bronchitis	Dyspnea
VisitToAsia	0	**0.9**	0	0	0	0	0	**0.9**
Smoking	0	0	0	0	0	0	**1**	0.1
Tuberculosis	0	0.1	0	0.55	**1**	0	**0.9**	0
LungCancer	0	**0.9**	0	0	0.45	0	0.1	0.05
TbOrCa	0	0.1	0	0.55	0	**1**	0	**0.95**
XRay	0	0	0	0	0	0	0	0
Bronchitis	0	0	0	0	0	0	0	**0.9**
Dyspnea	0	0	0	0.1	0	0	0.1	0

Table 4.2 Learning statistics of SA search

The comparison of these two tables shows that there are more edges learned as a result of Greedy search than SA search. The edges learned from Greedy search are associated with a low confidence level – most of them are below 0.5 and no one is above 0.8, while SA search produced 9 edges with confidence level above 0.8 (in bold). Among these high confidence level edges, some of these edges are confirmed by the original structure including "Smoking"->"Bronchitis", "Tuberculosis"-> "Tuberculosis or Cancer" ("TbOrCa"), "TbOrCa"->"XRay", "TbOrCa"->"Dyspnea" and "Bronchitis"->"Dyspnea", and others are inconsistent with the original structure, such as "Visit To Asia"->"Smoking" and "Lung Cancer"-> "Smoking". There are also a number of edges that were not discovered by SA search. Note that the overall results from both search methods are not satisfactory, but SA search over-performed Greedy search.

To include the pre-ordered approach into the evaluation, we used the number of correctly and incorrectly learned edges in this approach compared to the average calculated from statistics of Greedy and SA approaches. The comparison is given in Table 4.3.

Learning methods / Evaluation categories	Greedy	SA	Pre-Ordered
Correct edges	3.55	5.3	7
Missing edges	2.45	1.15	1
Reversed edges	2	1.55	0
Added edges	5.85	3.7	3

Table 4.3 Comparison of learned edges between Greedy, SA and Pre-ordered search

Pre-order search generates the highest number of correctly identified edges, lowest number of missing or added edges, and no reverse edge. This result is not surprising since the variables in the dataset are organized in such a way that a child variable is usually placed after its parent variable. Pre-ordered search is supposed to identify good network from such dataset very efficiently.

4.1.2 Influence of sample size on search performance

The previous experiment showed a higher efficiency of SA search over greedy search, but the average result learned with SA search shown in Table 4.2 is still far from being satisfactory. This is partly due to the fact that a small sample size was used in the search process. The previous experiment used a dataset with 1000 samples, which may not be large enough to extract sufficient information required to build an appropriate Bayesian network. The experiment in this section was to show the influence of the number of samples on learning performance.

The SA search method was used to learn Bayesian network for *Chest Clinic* dataset with 5000 and 50000 samples respectively. The statistics of the 20 runs are given in Table 4.4 and Table 4.5.

Children / Parents	VisitToAsia	Smoking	Tuberculosis	LungCancer	TbOrCa	XRay	Bronchitis	Dyspnea
VisitToAsia	0	0	0.7	0	0	0	0	0
Smoking	0	0	0	0	0	0	**0.8**	0.1
Tuberculosis	0.3	0	0	0.3	**1**	0	0.2	0.05
LungCancer	0	**1**	0	0	0.7	0	0	0.05
TbOrCa	0	0	0	0.3	0	**1**	0.2	**0.95**
XRay	0	0	0	0	0	0	0	0
Bronchitis	0	0.2	0	0	0	0	0	**0.8**
Dyspnea	0	0	0	0.1	0	0	0.2	0

Table 4.4 Learning statistics of SA search with 5000 samples

Children / Parents	VisitToAsia	Smoking	Tuberculosis	LungCancer	TbOrCa	XRay	Bronchitis	Dyspnea
VisitToAsia	0	0	0.65	0	0	0	0	0
Smoking	0	0	0	0.05	0	0	**0.95**	0
Tuberculosis	0.35	0	0	0.55	**0.85**	0	0	0.15
LungCancer	0	**0.95**	0.15	0	0.4	0	0	0.15
TbOrCa	0	0	0.15	0.6	0	**1**	0	**0.85**
XRay	0	0	0	0	0	0	0	0
Bronchitis	0	0.05	0	0	0	0	0	**1**
Dyspnea	0	0	0	0.1	0	0	0	0

Table 4.5 Learning statistics of SA search with 50000 samples

Compared to the results learned from the dataset of 1000 samples, the results with bigger sample size improved markedly in terms of the number of incorrectly learned edges. Using the threshold of 0.8 for confidence level, the number of incorrectly learned edges decreased by three. Only one edge (i.e. "Lung Cancer"->"Smoking") was incorrectly learned. Figure 4.1 shows the network learned from the 5000 samples with threshold of 0.8 confidence level. There are two missing edges, improved by two as compared with the network learned from 1000 samples.

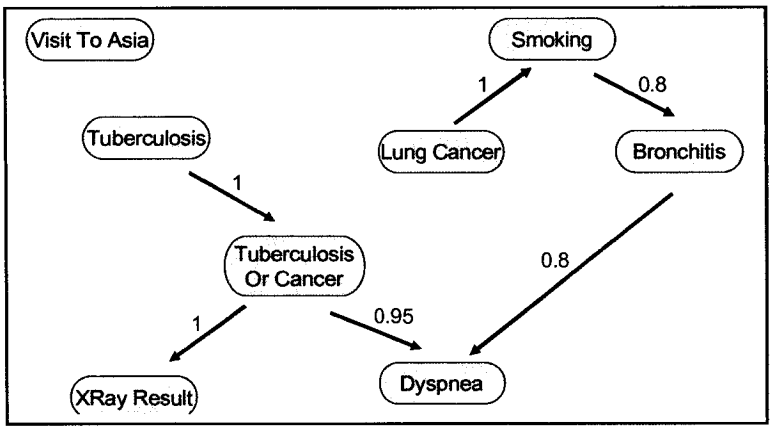


Figure 4.1 Learned network structure with 5000 samples

Table 4.6 shows the comparison of the statistics results from the three experiments (sample size 1000, 5000, and 50000). Performance gain is clear when increasing the sample size from 1000 to 5000. One interesting thing is that the experiment with 50000 samples does not perform better than the experiment with 5000 samples. This may be due to the variety of optimum points to which the search method can converge. This also tends to indicate that continuing increasing samples would not help to further improve the learning performance.

Evaluation categories \ Sample size	1000	5000	50000
Correct edges	5.3	5.95	5.75
Missing edges	1.15	0.05	0.15
Reversed edges	1.55	2	2.1
Added edges	3.7	0.8	1

Table 4.6 Comparison of learning performance. The value in each cell is an average of 20 runs.

4.1.3 Pre-knowledge on learning performance

We noticed that the learning performance is still not satisfactory even with increased sample size. This indicates that there are other factors besides sample size contributing to the learning performance. We also noticed that the wrong edge ("Lung Cancer"->"Smoking") of the learned network structure in Figure 4.1 has a high confidence level. This causal relationship is always reversed no matter what sample size is chosen. By examining the probability distribution of the dataset (Table 3.1), we observed that there is no difference in the number of people who smoke and who do not smoke (each with

50%). But there is a huge difference between the number of people who have cancer and who do not have. It seems that the score function used to learn Bayesian network chooses the variable that has larger difference between their distinct instantiations as parent, and in this case, the variable "Lung Cancer" is more likely selected as parent of the variable "Smoking".

To solve such problem in the learning process, knowledge about the related dataset should be used to impose certain restriction on the learning process. As we are sure that the variable "Smoking" in *Chest Clinic* dataset is a dominant variable that should not have any parent, the learning process should avoid attempting to search for parent variable of variable "Smoking". A parent candidate list can be defined to contain all the possible parent candidates for each variable and the learning process search for the potential parent for that variable only from the list. The parent candidate list of the variable "Smoking" should obviously contain no member. Also it is certain that "Visit To Asia" is another dominant variable and should not have any parent. With this restriction, we conducted the experiments for a dataset with 5000 samples and obtained the different results shown in Table 4.7:

Children / Parents	VisitToAsia	Smoking	Tuberculosis	LungCancer	TbOrCa	XRay	Bronchitis	Dyspnea
VisitToAsia	0	0	**0.85**	0	0.05	0	0.05	0
Smoking	0	0	0	**0.8**	0.2	0	**1**	0.1
Tuberculosis	0	0	0	0.2	**0.95**	0	0	0.3
LungCancer	0	0	0.05	0	**0.8**	0	0	0.3
TbOrCa	0	0	0.05	0.2	0	**1**	0.1	0.7
XRay	0	0	0	0	0	0	0	0
Bronchitis	0	0	0	0	0	0	0	**0.9**
Dyspnea	0	0	0	0.1	0	0	0.1	0

Table 4.7 Learning result with pre-knowledge applied for dataset with 5000 samples

The network with edges that have confidence level above 0.8 is shown in Figure 4.2. The improvement is significant: there are seven correctly learned edges, one missing edge and no incorrect edge.

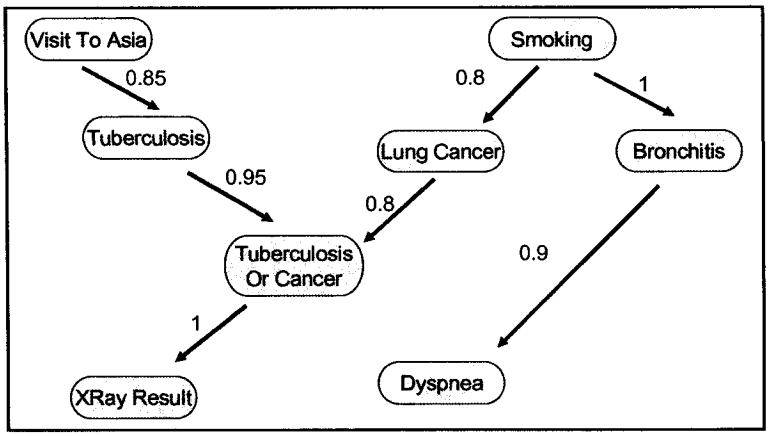


Figure 4.2 Learned network with pre-knowledge applied for dataset with 5000 samples.

4.1.4 Handling missing data

To determine the effect of missing data on the learning performance, we first simulated three datasets that contained 2%, 5% and 10% missing data, respectively based on the *Chest Clinic* data. A series of learning experiments were performed using the three datasets. Then we ran the same experiments with the KNN imputed data (see section *3.5.1 Imputation*). The dataset with 5000 samples was used and the SA search was applied using pre-knowledge. For each missing percentage, learning was performed for the dataset with missing data ignored (not handled) and with missing data imputed by KNN. The statistics of the learning results are shown in Table 4.8 along with the results from a dataset without any missing data.

Missing percentage / Evaluation categories	No missing	2% missing		5% missing		10% missing	
		No imputation	KNN	No imputation	KNN	No imputation	KNN
Correct edges	7	6.8	7.2	6.15	7.7	5.7	7.5
Missing edges	0.65	0.75	0.45	1.45	0.25	1.85	0.2
Reversed edges	0.35	0.45	0.35	0.4	0.05	0.45	0.3
Added edges	1.35	1.9	2.5	1.25	2.2	2.55	3.35

Table 4.8 Learning results before and after KNN imputation for different missing percentages

It can be seen from Table 4.8 that when there is missing data in the dataset, learning results are deteriorated if the missing data is not handled. As the missing percentage increases, the number of correctly learned edges is lower compared to the case of "No

missing" (6.8 for 2%, 6.15 for 5% and 5.7 for 10% missing); the number of missing edges is increased (0.75 for 2%, 1.45 for 5% and 1.85 for 10% missing); the number of reversed edges is increased (0.45 for 2%, 0.4 for 5% and 0.45 for 10% missing) and the number of added edges is increased except the one with 5% missing (1.9 for 2%, 1.25 for 5% and 2.55 for 10% missing). When KNN is used to impute the missing data, the learning results are improved. The number of correct edges is maintained at the same level compared to the case of "No missing", 7.2 for 2%, 7.7 for 5% and 7.5 for 10% missing, which is even better than the case without missing data. The reason is that the KNN algorithm attempts to impute missing with "similar" data, which results in a complete dataset with better pattern and less noisy information. The same improvements can be seen for missing edges and reversed edges as well. Only the added edges have their number increased. Figure 4.3 gives a histogram of the comparison for these learning results.

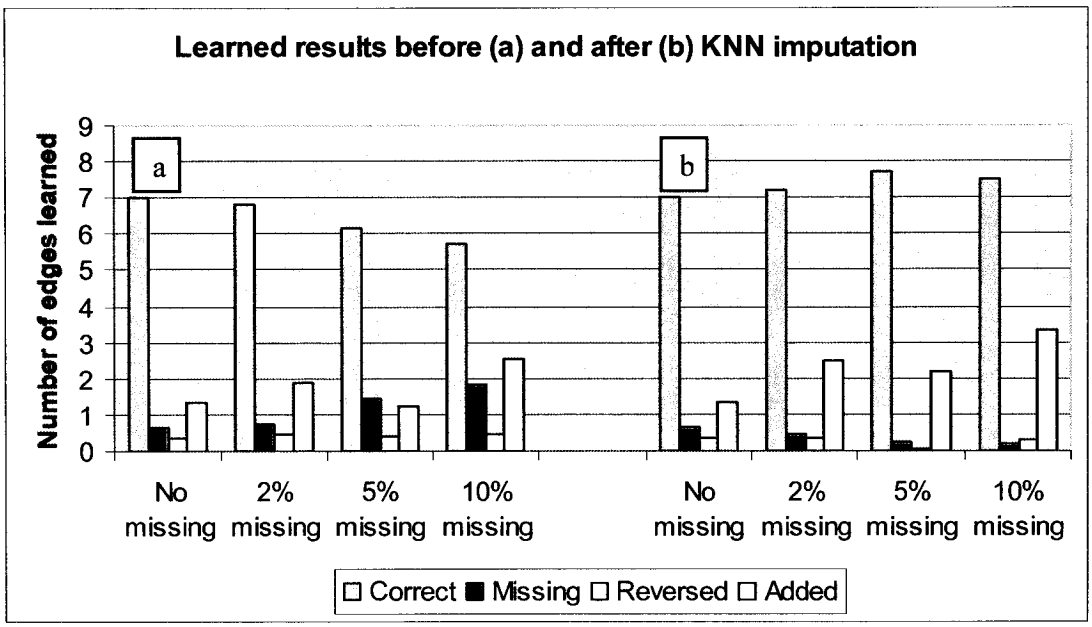


Figure 4.3 Learning results: a) without handling missing data the correct learned edges are decreased when the percentage of missing increased and b) with imputed data, the correct learned edges are increased.

It is worth to note that the presence of missing data can lead to different learning results. This is quite dependent on the missing pattern. The KNN imputation method seems to have limitation in reducing the impact of missing on Bayesian network learning – as

shown in Figure 4.3, the number of added edges is still high with this method (with imputed data). We can use other approaches, which can take into account possible parent-child relationship among the variables and the corresponding Bayesian probabilities. This will be further discussed in the conclusion section of this thesis.

4.2 Learning Bayesian network from microarray data

A major objective in post-genomic era is to fully understand the transcriptional control of each gene and the targets of each transcription factor. In yeast, large-scale experimental and computational approaches have been applied to identify co-regulated genes, *cis*-regulatory elements, and transcription factor DNA binding sites *in vivo*. Methods for modeling and predicting system behavior, and for reconciling discrepancies among data types, are being explored [Chua et al., 2004].

Yeast is a unicellular eukaryote. Its cell cycle is very much similar to human. Yeast is very easy and rapid to grow and mutate, it is also relatively well studied and reported in the literature. Therefore, yeast is an ideal model system for studying cell cycle biology.

Cell division cycle can be divided into four major phases [Weaver 1999]: G1 or gap 1, S for synthesis, G2 for gap 2 and M for mitosis or physical cell separation which are the traditional subdivisions of the cell cycle (see Appendix B). The most significant phases are the S phase (synthesis), during which the chromosomes are replicated and the M (cytokinesis) phase, during which the cell divides into two daughter cells.

4.2.1 Data presentation

A Bayesian network is learned from subset of a well-known yeast cell cycle dataset that was originally collected by Spellman et al [1998]. Data was retrieved online at: http://cellcycle-www.stanford.edu. This dataset contains gene expression measurements of the mRNA levels of 800 genes (ORFs- open reading frame) at 76 time points. The 800 genes were identified as cell-cycle related genes by Spellman et al. [1998]. Each of the 76 measurements will be considered as an independent sample data by ignoring the time influence on the experiment. The learning process will then try to find the relationships among subset of those 800 genes.

This dataset is standardized by the original authors by using a correlation function that identified genes whose RNA levels were similar to the RNA levels of the genes already known to be regulated by the cell cycle [Spellman et al., 1998].

Preliminary processes for the dataset were performed, for dealing with the missing data and the discretization of expression level.

- Handling missing data: KNN method was used to impute the missing values in the dataset, as it was shown that this method is simple and efficient [Troyanskaya et al., 2001].
- Data discretization: we used the simplest approach - the continuous gene expression levels were discretized into three discrete values: 1 for over-expressed; -1 for under-expressed and 0 for not-expressed (see section *2.1 Basic concepts*). Also the genes that do not have any expression during all the experiments (with discrete value 0) were filtered out since they do not provide any useful information.

The major problem with this dataset is that the dataset has a large number of genes (800) and only a small number of samples (76). It has been shown previously (see section *2.4 Challenges of microarray data*) that the sample size is an important part of the Bayesian Learning process – small sample size usually results in wrong networks because of the lack of information provided by the dataset. Some kind of pre-processing must be done for this dataset before applying the learning process in order to ensure the correct learning performance. Fortunately there is some pre-knowledge of the yeast dataset that we can rely on to make the required search space smaller. Basically this can help assign parent candidate list for individual genes to an appropriate and smaller set of genes, and the search can be restricted within the parent candidate list only.

4.2.2 Applying pre-knowledge

4.2.2.1 Subcellular localization information

The subcellular localization information indicates a cellular component where a gene product (protein) is located. Subcellular localization is a key functional characteristic of proteins. To co-operate for a common physiological function (metabolic pathway, signal transduction cascade, structural associate etc.), proteins must be localized in the same cellular compartment [Eisenhaber and Bork, 2000]. Huh and co-workers [Huh et al.,

2003] made 22 protein localization assignments, where 75% of the yeast proteome were assigned into 22 distinct subcellular localization categories by using the GFP tag and co-localization with RFP-tagged reference proteins. The published localization data can be found at http://yeastgfp.ucsf.edu. Figure 4.4 shows the defined locations in a cell. Analysis of this high-resolution, high-coverage localization dataset in the context of transcriptional, genetic, and protein–protein interaction data helps reveal the logic of transcriptional co-regulation, and provides a comprehensive view of interactions within and between organelles in eukaryotic cells.

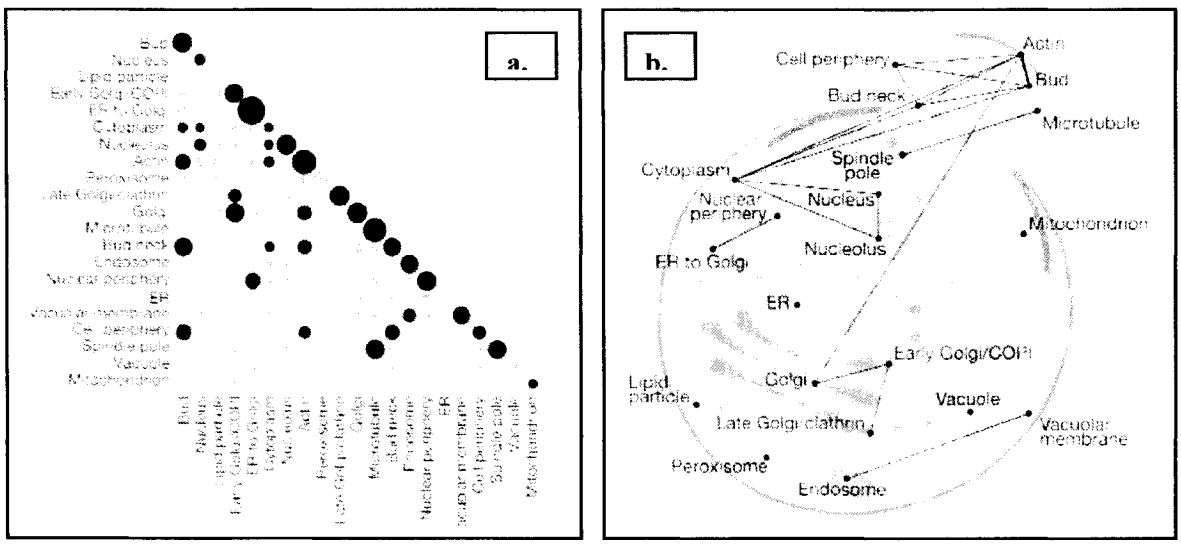


Figure 4.4 ***(Image credit: Huh et al, 2003)*** **a) Localization pairs were extracted from the localization of interacting partners. b) Diagram of the off-diagonal, statistically significant interacting compartments in a.**

4.2.2.2 Sub-networks

The pre-knowledge presented in previous sections allows using a smaller sample size to learn Bayesian networks. However compared to the large number of genes involved in the microarray experiments (about 800), the sample size (only 76 samples) is still too small. Establishing a network for all the 800 variables requires a huge amount of information within a search space that the learning algorithm can look up to find some reasonable networks. The information provided within these 76 samples in this dataset is far less than what would be required. In addition, learning a Bayesian network for a large number of variables is very resource consuming because the search space increases exponentially. The time and memory required to learn for the 800 genes in our dataset would not be satisfied by the current computing equipment.

Since applying the learning algorithm for all the 800 genes is not feasible, the alternative would be dividing the entire network into some smaller sub-networks. Fortunately sub-networks do exist in the yeast dataset and each of them has different function. These sub-networks have been reported by researchers and some of their functions have been confirmed [Friedman et al., 2000].

4.2.3 Learning results

Two groups of genes (CDC5 and CLN2) were selected according to the cell cycle information to learn their regulatory networks. One group (CDC5) of genes is selected from G2/M phase, which contains 7 genes. The second group (CLN2) is selected from G1 phase, which contains 7 genes. Part of the network for these genes has been reported by Friedman et al [2000]. Our objective was to validate our learning approach by comparing our result with the published regulatory network, and we specifically focused on the improvement of the learning performance by applying the pre-knowledge. Our learning also revealed some gene relationships that have not been reported previously. We analyzed those novel discovered relationships based on biological literature and biological background knowledge.

4.2.3.1 CDC5 data group

We begin with the learning without pre-knowledge applied. The statistics result from all the 20 runs are collected in Table 4.9. This table shows that there are too many edges produced from the learning (total 22 edges learned), and almost every gene is found to be the parent of at least one other gene (too interconnected). In addition, most of these learned edges have confidence level far below 0.8. In fact, there are only three of them that have confidence level higher than 0.8: CLB1->ALK1, CDC5->CLB2 and CLB2->SWI5. As we mentioned previously (see section *4.1.2 Influence of sample size on search performance*), this is due to the fact that the sample size is far less than the number of genes (sample size << number of genes). Because of this, many potential network structures and the learning process were unable to converge to the best one during most runs, which resulted in many scattered edges with low confidence level.

Children / Parents	SWI5	ALK1	CLB1	MYO1	ACE2	CDC5	CLB2
SWI5	0	0	0	0	0	0	0
ALK1	0	0	0.2	0	0	0.05	0
CLB1	0	**0.8**	0	0	0.45	0.5	0.7
MYO1	0.75	0.05	0	0	0	0.35	0
ACE2	0	0.25	0.5	0	0	0.5	0
CDC5	0.25	0.7	0.2	0.65	0.5	0	**0.85**
CLB2	**1**	0.05	0.3	0	0	0.15	0

Table 4.9 Learning statistics of CDC5 network without pre-knowledge applied

The statistics of the learning results for running the same experiment when applying pre-knowledge is collected in Table 4.10. It is obvious that by using pre-knowledge, the learning can only focus on the eligible relationships restricted by the pre-knowledge, thus greatly reducing the number of learned edges (15 in total). More importantly, there are more learned edges that have confidence level higher than 0.8 (7 edges). The network consisting of high confidence level edges is displayed in Figure 4.5.

Children / Parents	SWI5	ALK1	CLB1	MYO1	ACE2	CDC5	CLB2
SWI5	0	0	0	0	0	0	0
ALK1	0	0	0	0	0	0	0
CLB1	0	0	0	0	0.5	0.6	**0.85**
MYO1	0	0	0	0	0	0.15	0
ACE2	0	0	0.75	0	0	**0.85**	0
CDC5	**1**	0	0.15	**0.85**	0.15	0	**0.8**
CLB2	**1**	**1**	0.15	0	0	0.15	0

Table 4.10 Learned result of CDC5 gene group with pre-knowledge applied

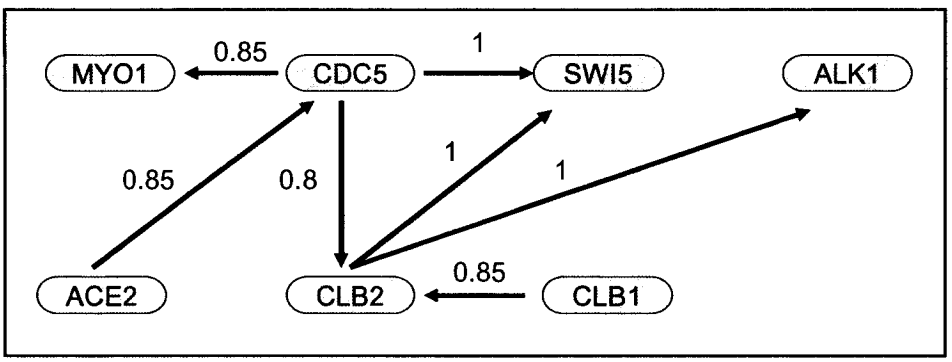


Figure 4.5 Learned network for CDC5 gene group with pre-knowledge applied

It is known that gene CDC5 encodes a protein that is involved in regulation of DNA replication and is the dominant gene in this sub-network [Friedman et al., 2000]. We uncovered three such relationships: CDC5->SWI5; CDC5->CLB2; and CDC5->MYO1, which have been published by Friedman [2000] and can be validated using biological knowledge. For example, CDC5 is found with confidence level of 1 as parent of SWI5. It is known that SWI5 encodes a transcription factor that activates transcription of genes expressed at the M/G1 boundary and in G1 phase of the cell cycle [Dolinski et al., 2003]. It is known that accumulation of CDC5 protein facilitates the activation of CLB2 protein proteolysis [Shirayama et al., 1998]. CDC5 functions in the pathway leading to the degradation of mitotic cyclin CLB2 [Song and Lee, 2001]. It is biologically reasonable that CDC5 is the parent of CLB2.

It is discovered that CDC5 has one parent gene ACE2 and this relationship has been reported by Friedman and colleagues [Friedman et al., 2000]. ACE2 encodes a transcription factor that activates transcription of genes expressing in G1 phase for the cell cycle.

Some relationships that have been published previously are not appearing in our learned network, such as CDC5->CLB2. We believe that this is due to the problem with the small sample size.

The network shown in Figure 4.5 reveals some relationships that were not discovered by Friedman [Friedman et al., 2000]. The most interesting is the relationship CLB2->SWI5. Transcription of both SWI5 and CLB2 is G2/M phase specified and activated in G2 phase [Nasmyth et al., 1987 and Koranda et al., 2000].

The product of SWI5 also functions as an important transcriptional regulator for cell cycle specific events that takes place in late anaphase and during G1 [Lydall et al., 1991 and Knapp et al., 1996]. The SWI5 gene has served as the archetypical G2/M specific transcriptional entity in yeast. Its cell cycle regulatory pattern closely mimics the CLB2 pattern and its promoter has been extensively used to identify and characterize G2/M – specific regulatory constituent [Koranda et al, 2000]. This relationship is strongly

suggested by our result with confidence level equal to 1. Figure 4.6 demonstrates the synchronization in gene expression between these two genes.

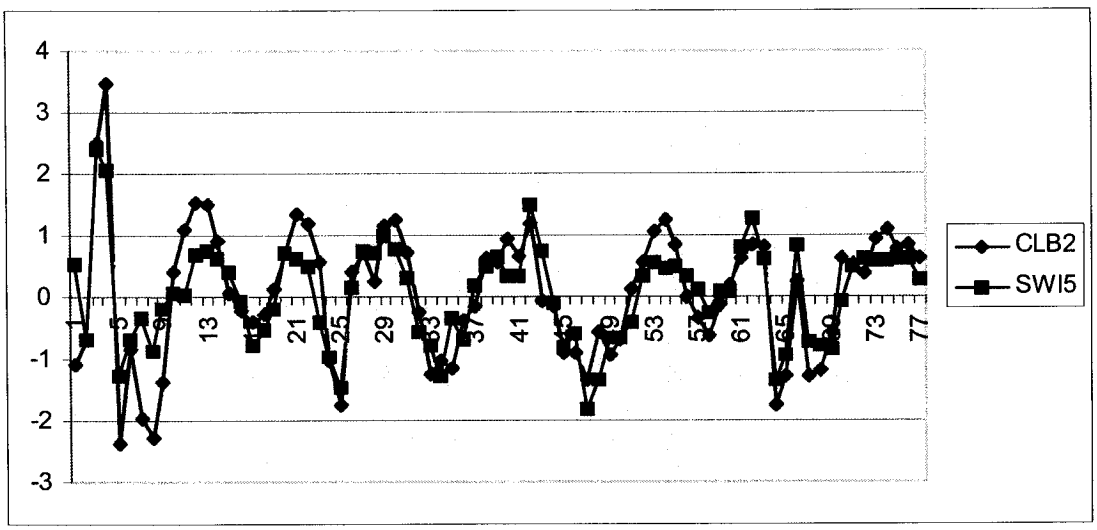


Figure 4.6 X-axis represents experiments (time points), Y-axis represents the fold change

Another interesting relationship is between CLB1 and CLB2 gene. These two genes are regulated by a combination of two transcription factors, Mcm1p and SFF [Spellman et al., 1998]. Both CLB1 and CLB2 encode B-type cyclin that activates Cdc28p to promote the transition from G2 to M phase of the cell cycle. CLB1 and CLB2 transcripts accumulate during G2 and M, and their transcription is repressed by the end of mitosis. They could have some influence on one another. Although this directed edge CLB1->CLB2 has not been reported previously, our learning results strongly suggest this relationship. Figure 4.7 shows the correlation of the expression profiles of these two genes.

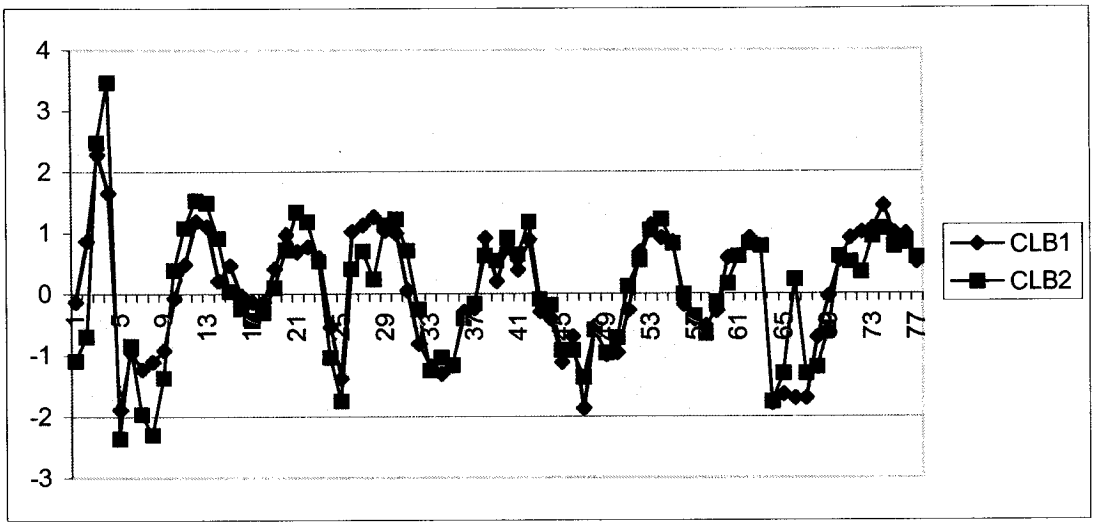


Figure 4.7 X-axis represents experiments (time points), Y-axis represents the fold change

4.2.3.2 CLN2 data group

The same (as CDC5 sub-network) learning strategies were taken for this group of data; two experiments were performed for CLN2 sub-network: with and without pre-knowledge. The statistics of the experiments are collected in Table 4.11 and Table 4.12. Similar to CDC5 sub-network, learning without pre-knowledge produces more edges between genes and confidence levels are usually low (Table 4.11) – there are a total of 16 edges and 4 of them have confidence level above 0.8. Applying pre-knowledge to the learning reduced the number of edges, resulting in more edges with higher confidence level (5 of total 12 edges have confidence level above 0.8) (Table 4.12). The learned network, with edges that have confidence level greater than 0.8, is shown in Figure 4.8.

Children / Parents	MNN1	RAD51	RNR3	SRO4	CLN1	SVS1	CLN2
MNN1	0	0	0	0	0	0	0
RAD51	0.25	0	0	0.25	0	0	0.15
RNR3	0.05	0	0	0	0	0	0.65
SRO4	0	0.45	0	0	1	0	0.3
CLN1	0	0	0	0	0	1	0
SVS1	0.4	0	0	0	0	0	0
CLN2	0.7	0.15	0.45	0.7	1	1	0

Table 4.11 Learned results for CLN2 gene group without pre-knowledge

Children / Parents	MNN1	RAD51	RNR3	SRO4	CLN1	SVS1	CLN2
MNN1	0	0	0	0	0	0	0
RAD51	0	0	0	0.6	0	0	0.15
RNR3	0	0	0	0	0	0	0.2
SRO4	0	0.15	0	0	**1**	0	0.65
CLN1	0	0	0	0	0	0	0
SVS1	**1**	0	0	0	0	0	0
CLN2	0	0.1	**0.8**	0.35	**1**	**1**	0

Table 4.12 Learned results for CLN2 gene group with pre-knowledge applied

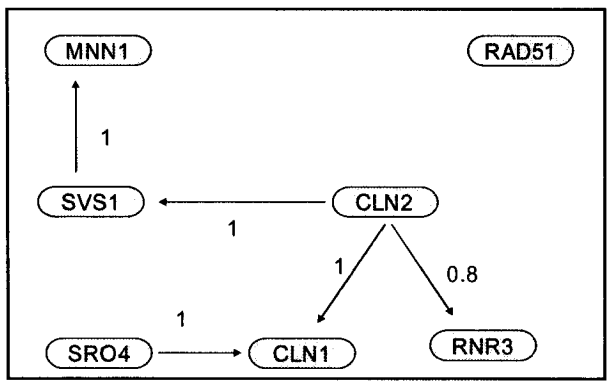


Figure 4.8 Learned network for CLN2 gene group with pre-knowledge applied

It is known that gene CLN2 is involved in early cell cycle, and controls all the other genes within this sub-network. Our learning revealed three genes that are regulated by CLN2: CLN1, RNR3 and SVS1. The learning discovered that gene MNN1 is regulated by gene SVS1. The other three regulations by CLN2 for genes RAD51, MNN1 and SRO4 reported by Friedman [Friedman et al., 2000], do no appear in our network.

We also discovered a relationship between SRO4 and CLN1 that were not previously reported. SRO4 appeared to regulate CLN1. Budding yeast propagates through budding. In this process, yeast cells become highly polarized and maintain their polarity as the bud emerges and grows, thus initiate the cell cycle. During this time, DNA is replicated and segregated. Once the bud has grown to roughly the size of its mother, cytokinesis occurs [Lord et al., 2000]. SRO4 (the encoded protein product is named Bud10p) is a trans-membrane protein that controls cell polarization during budding. This protein is localized at the incipient bud site in G1 phase and then remains in the mother–bud neck as the bud grows outward [Halme et al., 1996; Roemer et al., 1996]. CLN1 is involved in cell cycle

START, null mutant of CLN1 exhibits G1 arrest. These data led us to speculate that SRO4 might regulate CLN1 by transducing signal event through downstream GTPases cascade [Halme et al., 1996]. This information can assist researchers to design new experiments to validate this hypothesis in wet lab.

Chapter 5 Conclusions and Future Work

Bayesian network techniques have been used for discovering causal relationships among large number of variables in many applications. Gene regulation network is one such application where Bayesian techniques can be used to identify the interactions between a cluster of genes and individual genes. The most important challenge in applying Bayesian techniques to gene expression data is that the sample size is usually small and few relationships can be found with strong confidence of dependency from such dataset. This usually results in a network filled with many scattered relationships and few of them have high confidence level (see *Chapter 4*). To avoid this problem, we proposed a method that associates the Bayesian learning with a restriction list (or candidate list) that is made of domain knowledge, and the learning process only uses the relationships among the genes within the candidate. By focusing only on the eligible relationships, it is possible to reduce the number of learned relationships and discover structure with higher confidence level. Also, using a candidate list requires a smaller search space and greatly reduced learning time.

The proposed approach associated with several Bayesian network search methods have been implemented in this thesis. The implementation has been first tested using a toy dataset and evaluated by comparing learned results to the original network structure. Several learning parameters (such as different search methods, sample size) have been tested. The experiment results show that by using a pre-knowledge based candidate list, the learning performance was found with certain improvement.

The implementation has been applied for identifying gene network for a yeast dataset [Spellman et al., 1998], by using a candidate list resulted from existing knowledge on the location of gene product. Because the sample size is very small, experiments were applied to the reduced number of genes grouped based on cell cycle information. The learned networks revealed many interesting gene relationships that are consistent with previously published results. However, some other relationships reported by other researchers were not revealed from our networks. This is due to the fact that the sample size was too small. Interestingly, our network discovered several novel relationships, which have not been reported previously. Through analysis of gene expression profiles

between the related genes and biological knowledge, we feel that such relationships are highly probable and merit further confirmation through wet lab experiments.

This thesis provides a basic framework for applying Bayesian network technique to identify gene regulation network. Many new techniques can be applied to improve this framework, which will be our future research work:

- The yeast dataset has records that are time based gene expression levels. A more appropriate way to build its network would be considering the temporal factor by using time variables in the network model. This means that one or more variables that represent time influence need to be included in the network, which results in the learned network that describes not only the relationship between genes but also provides a view on how genes affect each other along with time variation. These networks are often referred as dynamic network to distinguish them from static network.

  Researchers are interested in building networks that include time series information and some interesting research results have been published. However, building dynamic networks for gene dataset are more complicated by a number of issues. For example, it has to be determined how to select proper time points and how to establish time-based interaction between genes. The network model can use continuous gene expression level, which eliminates the side effect introduced by the discretization process.

- It has been determined that small sample size is a problem in building Bayesian network. The yeast dataset that is used in this thesis has a very small number of samples, which can only result in learned networks with scattered relationships associated with low confidence level. One approach is to collect more data samples to provide enough information to the learning algorithm. This approach requires huge number of experiments, which is not available at this time due to the cost. Another approach is to include some knowledge in the learning process to restrict the learning to eligible relationships, thus avoiding arbitrary searching among a larger search

space. This thesis has used subcellular location information. Other existing knowledge that has been validated can also be included into the learning process.

Another approach is to implement algorithms that allow building candidate relationships during the search process. A method has been developed by [Friedman., 2000], which allows establishing a parent candidate list for each node based on current learned network and the information extracted from the dataset.

- It has been shown that the existence of missing data in a dataset could result in poor learning performance. The approach used in this thesis is the simplest one that imputes missing data with KNN method. Other algorithms (such as EM) may be used to deal with missing data more efficiently for Bayesian learning.

Appendix A

Following is an example of the gene regulatory network adopted from http://doegenomestolife.org.

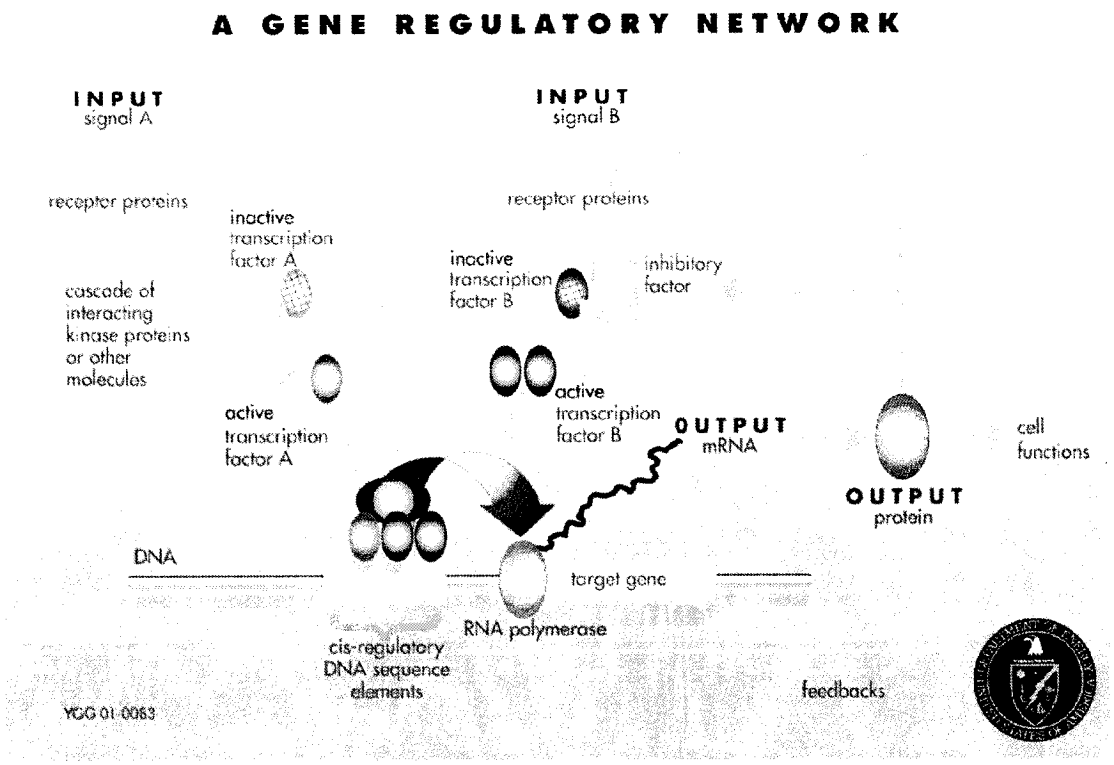
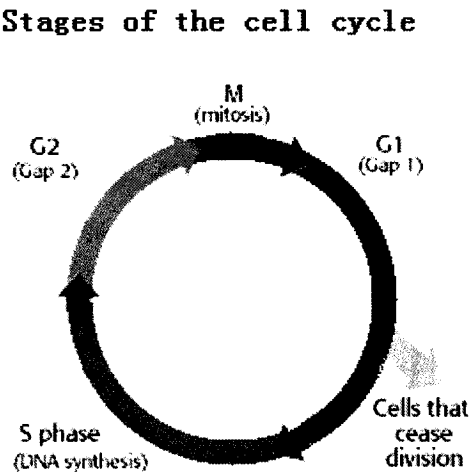


Image credit: U.S. Department of Energy Genomics: GTL Program, http://doegenomestolife.org. In this example, two different signals impinge on a single target gene where the cis-regulatory elements provide for an integrated output in response to the two inputs. Signal molecule A triggers the conversion of inactive transcription factor A (green oval) into an active form that binds directly to the target gene's cis-regulatory sequence. The process for signal B is more complex. Signal B triggers the separation of inactive B (red oval) from an inhibitory factor (yellow rectangle). B is then free to form an active complex that binds to the active A transcription factor on the cis-regulatory sequence. The net output is expression of the target gene at a level determined by the action of factors A and B. In this way, cis-regulatory DNA sequences, together with the proteins that assemble on them, integrate information from multiple signaling inputs to produce an appropriately regulated readout. A more realistic network might contain multiple target genes regulated by signal A alone, others by signal B alone, and still others by the pair of A and B.

Appendix B

(Adopted from http://www.biology.arizona.edu/cell_bio/tutorials/cell_cycle/cells2.html)



The cell cycle is an ordered set of events, culminating in cell growth and division into two daughter cells. There are four stages, G1-S-G2-M. Non-dividing cells not considered to be in the cell cycle.

G1 (Gap 1)	In this stage of the cell cycle where the cell is preparing to begin DNA replication.
S (Synthesis)	This is the stage when DNA replication occurs
G2 (Gap 2)	In this stage of the cell cycle, each chromosome is composed of two chromatids in preparation for mitosis. S resulted in the duplication of each chromatid. Since there is only one centromere on the sister chromatids, we still call them one chromosome. When completed, the cells are in G2 and preparing for M.
M (Mitosis)	This is the stage when nuclear (chromosomes separate) and cytoplasmic (cytokinesis) division occur.

References

Akutsu T., Miyano S. and Kuhara S. (1999) Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In *Proceedings of the 1999 Pacific Symposium in Biocomputing (PSB 99)*. pp. 17 28

Alam R, Gorska M: Genomic microarrays: arraying order in biological chaos? *Am J Respir Cell Mol Biol* 2001, 25:405-8.

Bar-Joseph Z., Gerber G., Gifford D., Jaakkola T. and Simon I., 'A new approach to analyzing gene expression time series data', *Proceedings of the Sixth Annual International Conference on Computational Biology*, 39-48, (2002).

Bar-Joseph Z, Gerber GK, Lee TI, Rinaldi NJ, Yoo JY, Robert F, Gordon DB, Fraenkel E, Jaakkola TS, Young RA, Gifford DK. Computational discovery of gene modules and regulatory networks. Nat Biotechnol. 2003 Nov; 21(11):1337-42. Epub 2003 Oct 12.

Cerny, V., "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm", *J. Opt. Theory Appl.*, 45, 1, 41-51, 1985

Chen T., Filkov V., Skiena S.S., Identifying Gene Regulatory Networks from Experimental Data. In proc. RECOMB, pages 94-103, 1999

Chen T., He H.L., Church G.M., Modeling Gene Expression with Differential Equations, Proc. Of Pacific Symposium on Biocomputing, pp.29-40, 1999

Chickering D. M., (1996), Learning Bayesian networks is NP-complete, in D. Fisher& H.-J. Lenz, eds, 'Learning from Data: Artificial intelligence and Statistics V', Springer Verlag.

Chua G., Robinson M. D., Morris Q., and Hughes T. R., Transcriptional networks: reverse-engineering gene regulation on a global scale. Current Opinion in Microbiology (ELSEVIER) 2004 Dec;7(6):638-46. Review

Cojocaru GS, Rechavi G, Kaminski N: The use of microarrays in medicine. *Isr Med Assoc J* 2001, 3:292-296.

Cooper G. F. and Herskovits F. A Bayesian Method for the Induction of Probabilistic Networks from Data', Machine Learning, 9. 309-347 (1992)

D'Haeseleer P., Wen X., Fuhrman S. and Somogyi R. 1999. Linear modeling of mRNA expression levels during CNS development and injury. *Pacific Symposium on Biocomputing '99*. Singapore: World Scientific. Pages 41 - 52.

Dolinski, K. et al. (2003). Saccharomyces Genome Database. http://www.yeastgenome.org

Duggan, D.J. et al. , 1999. Expression profiling using cDNA microarrays. Nature Genetics Supp. 21:10-14

Eisen M.B., Spellman P.T., Brown P.O., and Botstein D., 'Cluster analysis and display of genome-wide expression patterns', *Proc. Natl. Acad. Sci. USA*, 95, 14863-14868, (1998).

Eisenhaber F. and Bork P.,, "Subcellular Localization: Automatic analysis of SWISS-PROT annotations with Meta _ A(nnotator)", 2000. http://mendel.imp.ac.at/CELL_LOC/

Ellson J., Gansner E., Koutsofios E., North S., Woodhull G., Graphviz and Dynagraph: Statistic and Dynamic Graph Drawing Tools. *Graph Drawing Software; Springer-Verlag*, 2003

Everitt B., *Cluster Analysis*, Heinemann Educational Books, London, 1974.

Everitt B.S. and Dunn G., Applied Multivariate Data Analysis. 1992 Oxford University Press, New York, NY.

Famili A., Liu G., and Liu Z., 'Evaluation and optimization of clustering in gene expression data analysis', *Journal of Bioinformatics*, Volume 20, No.7, 2004.

Friedman, N., 'Learning belief networks in the presence of missing values and hidden variables'. In Proceedings of the Fourteenth International Conference on Machine Learning. (1997)

Friedman N., Linal M., Nachman I., and Pe'er D., Using Bayesian networks to analyze expression data. Journal of Computational Biology, 7(3/4): 601-620, 2000

Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coller H, Loh ML, Downing JR, Caligiuri MA, Bloomfield CD, Lander ES. Molecular classification of

cancer: class discovery and class prediction by gene expression monitoring. Science. 1999 Oct 15;286(5439):531-7

Grandin N and Reed SI (1993) Differential function and expression of Saccharomyces cerevisiae B-type cyclins in mitosis and meiosis. *Mol Cell Biol* 13(4):2113-25

Halme A., Michelitch M., Mitchell E. L., and Chant J., Bud10p directs axial cell polarization in budding yeast and resembles a transmembrane receptor. Current Biology 1996, Vol 6 No 5:570 579

Hartemink A. J, Gifford D.K, Jaakkola T.S., and Young R.A. Combining location and expression data for principled discovery of genetic regulatory models. In Pacific Symposium on Biocomputing, pages 437-449, 2002

Hartuv E., Schmitt A., Lange J., Meier-Ewert S., Lehrach H., and Shamir R., 'An algorithm for clustering cDNA fingerprints', *Genomics*, 66 (3), 249-256, (2000).

Huh W. K., Falvo J. V., Gerke L. C., Carroll A. S., Howson R. W., Weissman J. S. and O'Shea E. K., 'Global analysis of protein localization in budding yeast'. Nature, Vol 425 16 October 2003

Jain A.K. and Dubes R.C., *Algorithms for Clustering Data*, Prentice Hall, Englewood Cli.s, 1988.

Kaminski N, Friedman N: Practical approaches to analyzing results of microarray experiments. *Am J Respir Cell Mol Biol* 2002, 27:125-32.

Kim S.Y., Imoto S., and Miyano S., Dynamic Bayesian Network and Nonparametric Regression Model for Inferring Gene Networks. Genome Informatics 13: 371-372 (2002)

Kirkpatrick, S., C. D. Gelatt Jr., M. P. Vecchi, "Optimization by Simulated Annealing", *Science*, 220, 4598, 671-680, 1983.

Knapp, D., Bboite, L. Stillman, D. J. & Nasmyth, K. The transcription factor SWI5 regulates expression of the cyclin kinase inhibitor p40sic, mol, cell boil. 16, 5701-57-07, 1996

Kohonen T., *Self-Organization and Associative Memory*, Sringer-Verlag, 3rd edn., 1989.

Koranda M., Schleiffer A., Endler L., and Ammerer G., Forkhead-like transcription factors recruit Ndd1 to the Chromatin of G2/M-specific promotor. Letter to nature, 2000

Kraft P, Horvath S: The genetics of gene expression and gene mapping. *Trends Biotechnol* 2003, 21:377-8.

Kundaje A., Antar O., Jebara T., and Leslie C., 'Learning regulatory networks from sparsely sampled time series expression data', Technical report, Columbia University, (2002).

Kwon A.T., Hoos H. H. and Ng R. Inference of Transcriptional Regulation Relationships from Gene Expression Data. *Proceedings of the 18th ACM Symposium on Applied Computing (SAC 2003)*, 2003

Lew DJ, *et al.* (1997) "Cell cycle control in Saccharomyces cerevisiae." Pp. 607-695 in The Molecular and Cellular Biology of the Yeast Saccharomyces: Cell Cycle and Cell Biology, edited by Pringle JR, Broach JR and Jones EW. Cold Spring Harbor, NY: Cold Spring Harbor Laboratory Press

Liang S, Fuhrman S, Somogyi R (1998) REVEAL, A General Reverse Engineering Algorithm for Inference of Genetic Network Architectures. Pacific Symposium on Biocomputing 3:18-29.

Liang Y, Diehn M, Watson N, Bollen AW, Aldape KD, Nicholas MK, Lamborn KR, Berger MS, Botstein D, Brown PO, Israel MA. Gene expression profiling reveals molecularly and clinically distinct subtypes of glioblastoma multiforme. Proc Natl Acad Sci U S A. 2005, 102(16):5814-9

Lockhart D.J., Dong H., Byrne M.C., Follettie M.T., Gallo M.V., Chee M.S., Mittmann M., Wang C., Kobayashi M., Norton H and Brown E.L., 'Expression Monitoring by hybridization to high-density oligonucleotide arrays', *National Biotechnology*, Vol 14, pp. 1675-1680, (1996).

Lord M., Yang M. C., Mischke M. and Chant J., Cell Cycle Programs of Gene Expression Control Morphogenetic Protein Localization. The Journal of Cell Biology, Volume 151, Number 7, December 25, 2000 1501 1511

Luan Y. and Li H., Clustering of time-course gene expression data using a mixed-effects model with B-splines, Bioinformatics. 2003 Mar 1;19(4):474-82.

Lydall, D., Ammerer, G. &Nasmyth, K. A new role of Mcm1 in yeast: Cell cycle regulation of SWI5 transcription. Genes Dev. 5, 2405-2419, 1991

Metropolis N., Rosenbluth A., Rosenbluth M., Teller A., Teller E., "Equation of State Calculations by Fast Computing Machines", *J. Chem. Phys.*,21, 6, 1087-1092, 1953.

McQueen J., 'Some methods for classification and analysis of multivariate observation', *Proceedings of the 5th Berkeley symposium on Mathematical Statistics and Probability*, 271-297, (1967).

Michaels GS., Carr DB., Askenazi M., Fuhrman S., Wen X., Somogyi R.. Cluster analysis and data visualization of large-scale gene expression data. Pac Symp Biocomput. 1998; 42-53.

Nasmyth K., Seddon A. and Ammerer G. *Cell* cycle regulation of SW15 is required for mother-cell-specific HO transcription in yeast. Cell 1987 May 22; 49(4):549-58

Nutt C.L., Mani D. R., Betensky R.A., Tamayo P., Cairncross J.G., Ladd C., Pohl U., Hartmann C., McLaughlin M. E., Batchelor T. T., Black P. M., Deimling A.V., Pomeroy S.L., Golub T. R., and Louis D. N., Gene Expression-based Classification of Malignant Gliomas Correlates Better with Survival than Histological Classification. CANCER RESEARCH 63, 1602 1607, April 1, 2003

Ong I.M., Glasner J.D., and Page D., Modelling regulatory pathways in E. coli from time series expression profiles. Bioinformatics. 2002 Jul;18 Suppl 1:S241-8.

Ott S., Imoto S., Miyano S., Finding Optimal models for small gene networks. Pacific Symposium on Biocomputing 9:557-567(2004)

Ott S. and Miyano S., Finding Optimal Gene Networks Using Biological Constraints. Genome Informatics 14: 124{133 (2003)

Paul Jorgensen and Mike Tyers, The fork'ed path to mitosis. Genome Biology, 2000 1(3) review 1002.1-1002.4, 2000.

Pearl J., Probabilistic Reasoning in intelligent systems: Networks of Plausible Inference. Morgan Kaufman publishers, INC. 1988.

Pe'er D., Regev A., Elidan G. and Friedman N. Inferring subnetworks from perturbed expression profiles. Bioinformatics, pages S215-S224, 2001

Pearson K., On lines and planes of closest fit to systems of points in space, *Phil. Mag.*, 2 (1901), 559 572 London, 1974.

Ramoni M.F., Sebastiani P., and Cohen P.R., 'Bayesian clustering by dynamics', *Mach. Learning.*, 47, 9-121, (2002a).

Ramoni M.F., Sebastiani P., and Kohane I. S., 'Clustering analysis of gene expression dynamics', *Proc. Natl. Acad. Sci.USA*, 99, 9121-9126, (2002b).

Raychaudhuri S., Stuart J. M., and Altman R. B.. Principal Components Analysis to Summarize Microarray Experiments: Application to Sporulation Time Series. Pacific Symposium on Biocomputing 2000, Honolulu, Hawaii, 452-463. 2000.

Reinitz, J. and Sharp, D.H., 1995, *Mechanisms of Development*, 49, 133

Rifkin S.A. and Kim J., 'Geometry of gene expression dynamics', *Bioinformatics*, 18, 1176-1183, (2002).

Robinson, R.W. (1977). Counting unlabeled acyclic digraphs. *Lecture Notes in Math.* **622** (1977) 28-43

Schena, M., Shalon, D., Davis, R.W., Brown P.O.: "Quantitative monitoring of gene expression patterns with a complementary DNA microarray", *Science* **270**: 467-470, 1995.

Schliep A., Schönhuth A., and Steinhoff C., 'Using hidden Markov models to analyze gene expression time course data', *Bioinformatics*, 19 (Suppl. 1), i255-i263, (2003).

Segal E., Shapira M., Regev A., Pe'er D., Botstein D., Koller D. and Friedman. N.. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nat Genet. 2003 Jun; 34(2):166-76.

Sharan R. and Shamir R., 'CLICK: A clustering algorithm with applications to gene expression analysis', *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 8, 307-316, (2000).

Shirayama M., Zachariae W., Ciosk R. and Nasmyth K., The Polo-like kinase Cdc5p and the WD-repeat protein Cdc20p/fizzy are regulators and substrates of the anaphase promoting complex in *Saccharomyces cerevisiae.* The EMBO Journal Vol.17 No.5 pp.1336 1349, 1998

Someren, E.V., Wessels, L., and Reinders, M., Linear modeling of genetic networks from experimental data, *Bioinformatics*, 18:355 366, ISMB2002.

Song S. and Lee K. S., A Novel Function of *Saccharomyces cerevisiae CDC5* in Cytokinesis. The Journal of Cell Biology, Volume 152, Number 3, February 5, 2001 451 469 http://www.jcb.org/cgi/content/full/152/3/451

Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, Eisen MB, Brown PO, Botstein D, Futcher B. Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. Mol Biol Cell. 1998 Dec;9(12):3273-97

Spiegelhalter, David J. (1986). Probabilistic reasoning in predictive expert systems in Uncertainty in Artificial Intelligence, L.N. Kanal and J.F. Lemmer (Eds.), North-Holland, Amsterdam, pp. 47-67.

Tamayo P., Slonim D., Mesirov J., Zhu Q., Kitareewan S., Dmitrovsky E., Lander E.S., Golub T.R., 'Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation', *Proc. Natl. Acad. Sci. USA*, 96(6), 2907-2912, (1999).

Tatsuya A., Satoru K., Osamu M. and Satoru M. Identification of Gene Regulatoty Networks by Strategic Gene Disruptions and Gene Overexpressions. Proceedings of 9-th ACM-SIAM SODA, Pages 695-702, 1998

Tavazoie S., Hughes J.D., Campbell M.J., Cho R.J.and Church G.M., Systematic determination of genetic network architecture. Nature Genetics 22:281-285 July 1999

Terashima H., Fukuchi S., Nakai K., Arisawa M., Hamada K., Yabuki N., Kitada K., Sequence-based approach for identification of cell wall proteins in Saccharomyces cerevisiae. *Curr Genet* 40(5):311-6

Tian T. and Burrage K., Stochastic Neural Network Models for Gene Regulatory

Networks. Proceedings of the 2003 Congress on Evolutionary Computation, 162-169, IEEE Press, 2003.

Troyanskaya O., Cantor M., Sherlock G., Brown P., Hastie T., Tibshirani R., Botstein D., and Altman R. B., Missing value estimation methods for DNA microarrays. BIOINFORMATICS. Vol. 17 no.6 2001 Pages 520 525

Tzouvelekis A, Patlakas G and Bouros D. Application of microarray technology in pulmonary diseases. *Respiratory Research* 2004, 5:26

Velculescu V.E., Zhang L., Vogelstein B., and Kinzler K.W., 'Serial analysis of gene expression', Science, Vol. 270, pp. 484-487, (1995).

Wahde M. and Hertz J., Coarse-grained reverse engineering of genetic regulatory networks. *Biosystems*, 55:129 136, 2000.

Walker, P.R., Smith, B., Liu, Q.Y., Famili, F., Valdes, J., Liu, Z., Lach, B. Data Mining of Gene Expression Changes in Alzheimer Brain. Artificial Intelligence in Medicine, Elsevier Science. June 2003.

Ward J.H., 'Hierarchical grouping to optimize an objective function', *Journal of the American Statistical Association*, 58, 236-244, (1963).

Weaver R. F. Molecular Biological. M^{c} Graw-Hill (1999).

Wichert S., Fokianos K. and Strimmer K., Identifying Periodically Expressed Transcripts in Microarray Time Series Data. Bioinformatics. 2004 Jan 1; 20(1): 5-20.

William D. Stansfield, Jaime S. Colome and Raul J. Cano. (1996) Molecular and Cell biology

Witten I. H. and Frank E., Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations. 2000.

Yoo C., Thorsson V., and Cooper G.F. Discovery of causal relationships in a gene regulation pathway from a mixture of experimental and observational DNA microarray data. In Pacific Symposium on Biocomputing, pages 498-509, 2002.