The Hong Kong Polytechnic University
Department of Computing

Effective Techniques for Gene Expression Data Mining

Ma Chi Hung, Patrick

A thesis submitted in partial fulfilment of the requirements for the Degree of Doctor of Philosophy

February 2006

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____ (Signed)

MA CHI HUNG (Name of student)

Abstract

Gene expression data mining as a new research area poses new challenges to data mining researchers. Gene expression data are typically very noisy and have very high dimensionality. To tackle bioinformatics problems involving them, traditional data mining techniques may not be the best tools to use as they were not originally developed to deal with such data. For this reason, new effective techniques are required. In this thesis, we propose some such techniques.

In particular, these techniques can be used to address the problems of reconstructing gene regulatory networks and clustering gene expression data. The former is concerned with the problem of discovering gene interactions to infer the structures of gene regulatory networks. The latter is concerned with the problem of discovering clusters of co-expressed genes so that genes that have similar expression patterns under different experimental conditions can be identified.

To reconstruct gene regulatory networks, we have proposed to use an association-discovery technique, which is based on residual analysis and an information theoretic measure, to detect whether or not there interesting association relationships between genes. Given time-dependent gene expression data, this technique can reveal interesting sequential associations between genes for the effective inference of the structures of gene regulatory networks.

The association-discovery technique proposed can also be used to find interesting association relationships between gene expression levels and cluster labels. Based on discovering such relationships, we have developed a two-phase clustering algorithm for gene expression data. This algorithm consists of an initial clustering phase and a second re-clustering phase. Using this two-phase approach, it

is able to group genes, whose cluster memberships cannot be easily determined by existing methods, into the appropriate clusters. Since the effectiveness of the two-phase clustering algorithm depends, to some extent, on that of the existing clustering method used in the first phase, therefore, we have developed a novel evolutionary clustering algorithm, called EvoCluster, that can be used in the first phase to overcome some of the limitations of existing ones. By making use of an evolutionary approach and the association-discovery technique, it not only is able to perform well in the presence of very noisy data, it can also be used to discover overlapping clusters.

For performance evaluation, the data mining techniques proposed in this thesis have been tested with simulated and real data and the experimental results show that they are very promising.

Publications arising from the thesis

1. Ma, P.C.H., Chan, K.C.C., Yao, X. and Chiu, D.K.Y., "An evolutionary clustering algorithm for gene expression microarray data analysis," accepted by *IEEE Trans. on Evolutionary Computation*, to appear in 2006.
2. Ma, P.C.H., Chan, K.C.C. and Chiu, D.K.Y., "Clustering and re-clustering for pattern discovery in gene expression data," *Journal of Bioinformatics and Computational Biology*, vol. 3, no. 2, pp. 281-301, 2005.
3. Ma, P.C.H. and Chan, K.C.C., "A novel data mining algorithm for reconstructing gene regulatory networks from microarray data," *Proc. of the 21st Annual ACM Symposium on Applied Computing (ACMSAC06 – Bioinformatics)*, Dijon, France, Apr. 23-27, 2006, pp. 202-203.
4. Ma, P.C.H. and Chan, K.C.C., "An effective data mining technique for the discovery of gene regulatory networks from time-series expression data," Submitted for journal publication (*under review*).
5. Ma, P.C.H. and Chan, K.C.C., "Inference of gene regulatory networks from microarray data," *Proc. of the Fourth Asia Pacific Bioinformatics Conference (APBC06)*, Taipei, Taiwan, Feb. 13-16, 2006, pp. 17-26. Also, in the series on *Advances in Bioinformatics & Computational Biology*, vol. 3, Imperial College Press, 2006.
6. Ma, P.C.H. and Chan, K.C.C., "A clustering algorithm for discovering overlapping clusters from noisy gene expression data," Submitted for journal publication (*under review*).

7. Ma, P.C.H. and Chan, K.C.C., *Mining Overlapping Clusters in Gene Expression Data*, Technical Report, COMP-02-05, Department of Computing, The Hong Kong Polytechnic University, 2005.
8. Ma, P.C.H. and Chan, K.C.C., *Discovering Clusters in Gene Expression Data*, Technical Report, COMP-07-04, Department of Computing, The Hong Kong Polytechnic University, 2004.
9. Ma, P.C.H. and Chan, K.C.C., "Discovering clusters in gene expression data using evolutionary approach," *Proc. of the 15th IEEE International Conf. on Tools with Artificial Intelligence (ICTAI03)*, Sacramento, California, USA, Nov. 3-5, 2003, pp. 459-466.
10. Ma, P.C.H., Chan, K.C.C. and Chiu, D.K.Y., "Discovering clusters in gene expression data," *Proc. of the 7th Joint Conf. on Information Sciences (jointly 5th Atlantic Symposium on Computational Biology and Genome Informatics, CBGI03)*, Research Triangle Park, N. Carolina, USA, Sept. 26-30, 2003, pp. 879-882.
11. Ma, P.C.H. and Chan, K.C.C., "Clustering gene expression data with hybrid GA approach," *Proc. of the 7th IASTED International Conf. on Artificial Intelligence and Soft Computing (ASC03)*, Banff, Alberta, Canada, Jul. 14-16, 2003, pp. 223-228. Also in *Artificial Intelligence and Soft Computing*, ACTA Press, 2003.

Acknowledgements

First of all, I would like to express my sincere appreciation to my PhD supervisor, *Prof. Keith Chan,* Department of Computing, The Hong Kong Polytechnic University, for his patience and guidance.

I would like to thank *Prof. David Chiu,* Department of Computing & Information Science and a Graduate Faculty of the Biophysics Interdepartmental Group, University of Guelph, *Prof. Xin Yao,* School of Computer Science, University of Birmingham, *Prof. Anil Jain,* Department of Computer Science & Engineering, Michigan State University, and *Prof. Paul Wang,* Department of Electrical & Computer Engineering, Duke University, for giving me many valuable advices.

I would also like to thank *Prof. Yi Pan,* Department of Computer Science, Georgia State University, and *Prof. David W. Corne,* School of Mathematical and Computer Sciences, Heriot-Watt University, for giving me many valuable suggestions on the thesis revision.

And also, I would like to thank my colleagues and friends in the Department of Computing and the PolyU Student Halls of Residence for their friendship and encouragement.

Lastly, I would like to thank my parents, my sisters and *Miss Erica Lai*. Without their love and spiritual support, this thesis is not possible.

Table of contents

Chapter 1

Introduction

The study of bioinformatics is multi-disciplinary [32]. It combines several scientific disciplines including molecular biology, biochemistry, mathematics, and computer science. It is concerned with managing, analyzing and interpreting a huge volume of biological data such as DNA and protein sequence data, RNA and protein structural data, and gene expression data, etc. As the proliferation of genome sequencing projects have resulted in an exponential growth in the numbers and sizes of such databases, there has recently been an increasing demand for computing techniques to deal with such data. To cope with the demand, effective data mining techniques are required to extract interesting, nontrivial, implicit, previously unknown and potentially useful information from such data [68].

1.1 The Problems

The recent advent of DNA microarray technology [90], [132] has made possible the simultaneous monitoring of the expression levels of thousands of genes. Specifically, this technology provides biologists with the ability to measure the relative levels of mRNA abundance of thousands of genes between different samples or between different time points of the same sample. This development is having a significant impact on many areas such as in biomedicine and pharmacogenomics, etc. If hidden regularities can be discovered in such gene expression data generated by microarray technology, it can facilitate elucidation of the signatures of complex diseases and the

development of individually optimized drugs, etc [16], [27]. Given the potential benefit, there is, therefore, a growing demand for effective techniques to mine gene expression data so as to uncover biologically meaningful patterns hidden in them. In this thesis, we propose some such techniques. In particular, these techniques can be used to effectively reconstruct gene regulatory networks (GRNs) and cluster gene expression data.

A GRN is a complex biological system in which genes interact with each other indirectly via the proteins they create to perform various cellular processes. Given a set of time-dependent gene expression data, if the expression of a gene is dependent on the expression of another gene, one would expect to observe that the expression levels of a gene be associated with that of another after a certain amount of time delay [15]. For the reconstruction of GRNs, such time-dependent gene interactions need to be discovered first. If this can be done effectively, the structures of GRNs can be inferred directly from data and it will be possible for us to have better understanding of how cellular processes are carried out to accommodate changes in the external environment. The problem of mining time-dependent gene expression data for GRNs is therefore an important bioinformatics problem that we would like to focus on here.

The problem of discovering clusters of co-expressed genes in gene expression data is another problem of great importance in bioinformatics. Co-expressed genes are genes that have similar expression patterns under different experimental conditions and they may also have similar or related biological functions [16]. Clustering of gene expression data can therefore help understand the functions of genes for which biological information has not been previously available. Furthermore, a strong correlation of expression patterns between co-expressed genes

can be an indication that they might be co-regulated by the same transcription factors and might have common binding sites. Once co-expressed genes are identified, the promoter regions of their corresponding DNA sequences can be searched for common patterns. Transcription-factors binding sites specific to each cluster may then be identified among common patterns found in these co-expressed genes [163].

Over the past decade, many data mining techniques have been developed to tackle different problems in a variety of application domains and they have been shown to be very effective [99]. Due to the uniqueness of the data involved, gene expression data mining, however, poses new challenges to data mining researchers. In order to tackle the problems of reconstructing GRNs and discovering clusters of co-expressed genes from data of such nature, we also need to effectively handle the following two important challenges [143].

The first challenge comes from the presence of noise inherent in the data. Sources of noise in gene expression data include experimental, measurement, reporting and other data processing errors. Due to the complexity of biological systems, theoretical estimation of error level in the expression data is difficult. One way to make a fair estimate of the error level is by interviewing biologists who understand the experimental processes that generated those data. However, it is usually not possible to do this. In the absence of a better estimate of error level and in order for useful patterns to be discovered, there is a need to have a data mining technique that is able to handle noise well in gene expression data [18], [151], [153].

The second challenge is concerned with the need to deal with a large number of irrelevant attributes. Although irrelevant attributes are present in almost all kinds of databases, the ratio of irrelevant to relevant attributes is likely not as large as that in

gene expression data. In most of the expression data, the number of relevant genes, which are relevant for the determination of a particular class of disease, is usually very small when compared with the total number of genes. The presence of those irrelevant genes often interferes with the discrimination power of those that are relevant when existing data mining techniques are used. This can not only result in extra computational time in the data mining process, but also increase the difficulty level of the problem. To make matters worse, the samples that can be gathered are normally relatively small in size. This makes the problem of uncovering hidden patterns in gene expression data more difficult.

1.2 Overview of Solutions

For GRNs reconstruction, we have proposed to use an association-discovery technique [113]-[115], which is based on residual analysis and an information theoretic measure, to detect whether or not there exists interesting association relationships between genes. By computing an average gene expression value which serves as a reference point for how large the value is, the proposed technique can discover interesting sequential associations between genes such as “if a gene is highly expressed, its dependent gene is then lowly expressed in the next time point” etc. These findings can not only allow hidden regularities to be easily interpreted, they can also determine if a gene is supposed to be activated or inhibited and can be used to predict how a gene would be affected by other genes from the unseen samples.

Given clusters (or classes) of genes, the association-discovery technique proposed can also be used to construct classifiers by finding interesting association relationships between gene expression levels and cluster (or class) labels. Based on

discovering such relationships, we have developed a two-phase clustering algorithm [112], [118], [120] for gene expression data. This algorithm consists of an initial clustering phase and a second re-clustering (or re-classification) phase. In the first phase, an existing clustering algorithm, such as *k*-means or a hierarchical clustering algorithm, can be applied. The results, which consist of a number of initial clusters, can then be used for re-clustering. The re-clustering problem can be formulated as a classification problem by treating the data in each initial cluster as training data for the construction of a classifier. Once the classifier is constructed, the genes in each initial cluster can then be re-classified either into the same cluster or into different clusters. With this two-phase approach, the proposed algorithm is able to effectively determine cluster memberships of genes whose cluster memberships cannot be easily determined by existing methods.

Since the effectiveness of the two-phase clustering algorithm depends, to some extent, on that of an existing clustering method used in the first phase, we have developed a novel evolutionary clustering algorithm, called EvoCluster [111], [119], [121], that can be used in the first phase to overcome some of the limitations of existing ones. It not only is able to perform well in the presence of very noisy data, it can also be used to discover overlapping clusters [116]-[117]. EvoCluster makes use of an evolutionary approach to guide the search for optimal or near-optimal clustering arrangement. To do so, it encodes the entire cluster grouping in a *chromosome*[1] so that each *gene* encodes one cluster and each cluster contains the labels of the data records grouped into it. Then, given the encoding scheme, it has a set of special crossover and mutation operators that facilitates the exchange of

[1] The terms such as *chromosomes* and *genes*, when used in a computational context, may not have the same meanings as their biological counterparts. In order to avoid possible confusion, when referring to these terms in the contexts of evolutionary computation, they are made italic.

grouping information between two *chromosomes* on one hand and allows variation to be introduced to avoid trapping at local optima on the other. In addition, for fitness evaluation, EvoCluster makes use of the association-discovery technique to discover interesting association relationships in each possible cluster to estimate how good the cluster arrangement encoded in a *chromosome* is.

For performance evaluation, the data mining techniques proposed in this thesis have been tested with simulated and real data and the experimental results show that they are very promising.

1.3 Organization of Thesis

This thesis is organized into six chapters as follows. Chapter 2 introduces the background materials that include some basic concepts in molecular biology, the process of knowledge discovery in databases and some popular data mining tasks. Moreover, a survey of related work of this thesis is also presented in this chapter.

Chapter 3 introduces an association-discovery technique for the discovery of interesting association relationships between genes from time-dependent expression data so that the structures of gene regulatory networks can be better inferred.

In Chapter 4, a two-phase clustering algorithm is presented. Using this two-phase approach, it is possible to group genes, whose cluster memberships cannot be easily determined by existing methods, into the appropriate clusters.

Chapter 5 introduces a novel evolutionary clustering algorithm to clustering gene expression data. In addition, the extended version of the proposed evolutionary clustering algorithm is also presented. This extended version is able to discover

overlapping clusters so that genes that have similar expression patterns with different groups of genes can be revealed.

In Chapter 6, we conclude the thesis and end with directions for future work.

Chapter 2

Background and Related Work

2.1 Some Basic Concepts in Molecular Biology

Molecular biology is the study of biology at a molecular level [27]. This field overlaps with other areas of biology, such as genetics and biochemistry. Molecular biology mainly concerns itself with understanding the interactions between the various biological systems of a cell, including the interrelationships of DNA, RNA and protein synthesis and learning how these interactions are regulated.

2.1.1 DNA

The nucleus of a cell contains chromosomes that are made up of the double helical DNA molecules. The whole stretch of DNA is called the genome of the organism. The DNA consists of two strands of phosphate and deoxyribose sugar molecules, joined by covalent bonds. To each deoxyribose sugar molecule is attached one of the four nitrogenous bases, namely, adenine (A), cytosine (C), guanine (G), and thymine (T). Note that uracil (U) exists in place of thymine (T) in ribose sugar (for RNA). Bases between the strand pairs are attached by hydrogen bonds, such that either AT (or AU) or GC comes together. DNA in the human genome is arranged into 24 distinct chromosomes that are physically separate molecules ranging in length from about 50 million to 250 million base pairs and there are approximately 3 billion base pairs in our genome. A nucleotide is a combination of a phosphate, a sugar, and a

purine or a pyrimidine base, where a purine (pyrimidine) consists of A or G (C or T or U) (Fig. 2.1).

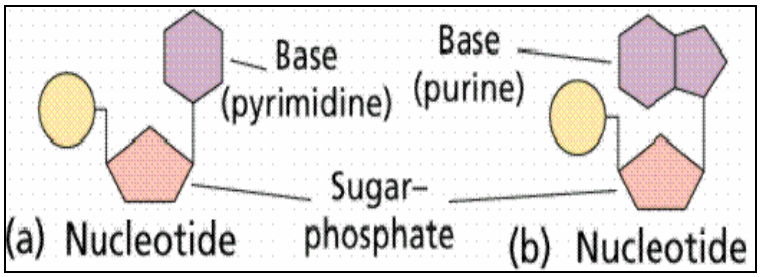

Fig. 2.1 Nucleotide is a combination of a phosphate, a sugar, and a purine or a pyrimidine base [62].

Genes, the basic physical and functional units of heredity, are coded in fragments of DNA (either strand) that are dispersed in the genome and each gene contains information to produce a protein. Understanding what parts of the genome encode which genes is one of the tasks of the Human Genome project [26]. Each chromosome contains many genes, however, genes comprise only a small portion of the human genome. At the time of writing, it is believed that there are only 20,000–25,000 genes in our genome [137]. The remainder consists of non-coding regions, whose functions may include providing chromosomal structural integrity and regulating where, when and in what quantity proteins are made. For protein production, each gene is first transcribed to produce messenger RNA (mRNA), which is then translated to produce protein. The mRNA is single-stranded and has a ribose sugar molecule. In humans a gene consists of exons that get translated into an amino acid sequence, separated by introns (that are not translated). There exist promoter and termination sites in a gene responsible for the initiation and termination of transcription. Translation consists of mapping from triplets (codons) of four bases to the 20 amino acids building block of proteins (Fig. 2.2). It should be noted that more than one triplet can map to the same amino acid, but the same triplet cannot map to different amino acids [154].

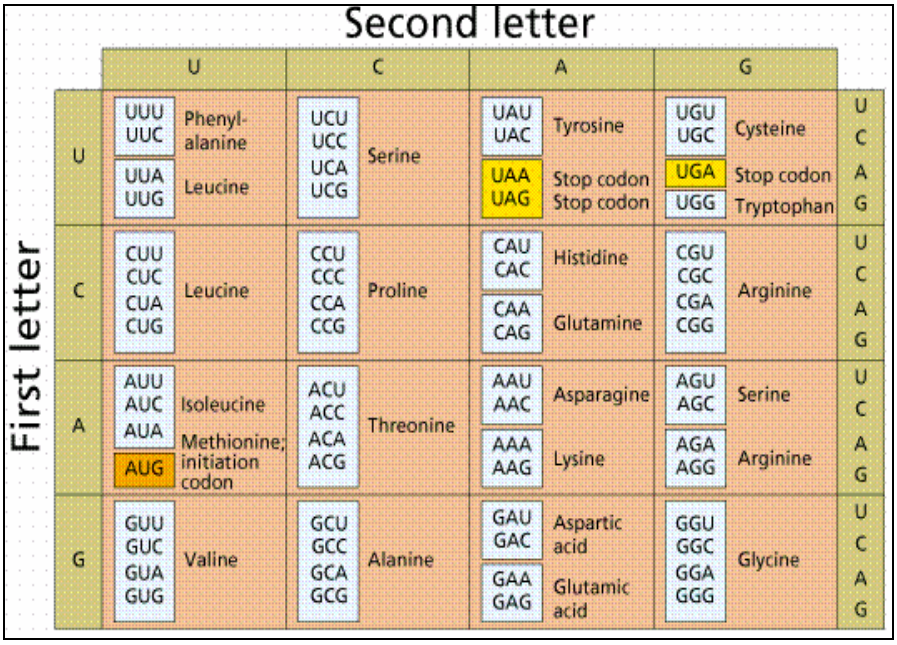

Fig. 2.2 The genetic code [62].

An amino acid is an organic molecule consisting of an amine and a carboxylic acid groups (backbone), together with a side chain (hydrogen atom and residue) that differentiates between them. The carboxyl and amino groups of a pair of amino acids react through hydrolysis (removal of a water molecule) to link and form a peptide bond. Similar reactions occur along the chain to form a protein molecule. A sequence of amino acids, held together by peptide bonds forming a polypeptide chain, endow a protein with its three-dimensional structure.

2.1.2 Proteins

Proteins perform most life functions and even make up the majority of cellular structures in our body. Proteins are polypeptides, formed within cells as a linear chain of amino acids. The length of a protein can vary from 10s to 1000s of amino acid monomers. Chemical properties that distinguish the 20 different amino acids cause the protein chains to fold up into specific three-dimensional structures that define their particular functions in the cell.

Proteins are involved in virtually every biological process in a living system. They are synthesized on ribosomes as linear chains of, typically, several hundred amino acids in a specific order from information encoded within the DNA. In order to function, these chains must fold into the unique native three-dimensional structures that are characteristic of the individual proteins. In a cell, this takes place in a complex highly crowded molecular environment. There are several families of proteins whose job is to catalyze the folding process of the other proteins that are required by the living organism. In a cellular environment, molecular chaperones help to protect the incompletely folded polypeptide chains from aggregating. Even after the folding process is complete, however, a protein can subsequently experience conditions under which it unfolds, at least partially, and then again becomes prone to aggregation. The failure of proteins to fold correctly or to remain unfolded under all appropriate physiological conditions can give rise to a wide range of pathological conditions, such as genetic, sporadic and even infectious ailments [27].

The constellation of all proteins in a cell is called its proteome. Unlike the relatively unchanging genome, the dynamic proteome changes from minute to minute in response to tens of thousands of intra- and extra-cellular environmental signals. A particular protein’s chemistry and behavior is specified by the gene sequence and by other proteins made in the same cell at the same time and with which it associates and reacts. Studies to explore protein structures and activities, known as proteomics, will be the focus of much research for decades to come and will help elucidate the molecular basis of health and diseases.

2.1.3 Gene Expression and DNA Microarray Technology

Every cell in an organism has the same set of chromosomes, but they can have very distinct properties. This is due to differences in the abundance, state, and distribution of cell proteins. The changes in protein abundance are in turn partly determined by the changes in the levels of mRNAs. The process of transcribing the gene's DNA sequence into mRNA that serves as a template for protein production is known as gene expression [154] (Fig. 2.3).

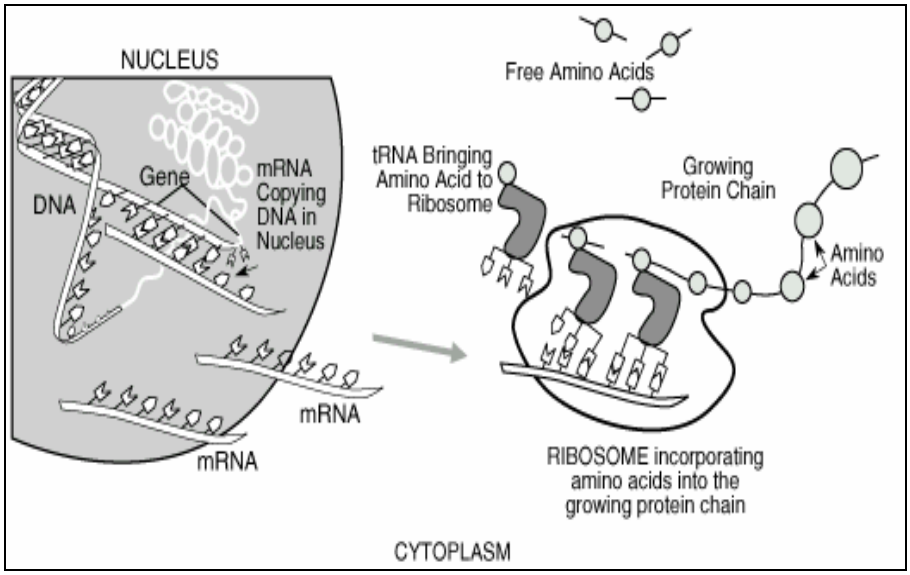

Fig. 2.3 Gene expression [62].

Gene expression consists of two basic steps: transcription and translation.

- *Transcription*: This entails the synthesis of a single-stranded RNA at an unwound section of DNA with one of the DNA strands serving as a template for the synthesis of the RNA. The product of this process is called a messenger RNA (mRNA) for protein genes or a functional RNA (tRNA or rRNA). The result of transcription is that the genetic information encoded in DNA is transferred to RNA, this occurs in the nucleus of the cell. For RNA genes, the expression is complete after a functional tRNA or rRNA is generated. However, protein genes require additional steps. The mRNA

carries this genetic information out of the nucleus and into the cytoplasm, where it becomes directly involved with protein synthesis via translation.

- *Translation*: This follows the movement of mRNA to the cytoplasm where it interacts with structures called ribosomes to synthesize a protein. Proteins are a linear sequence of amino acids, each of which is specified by the sequence of nucleotides in the RNA molecule.

DNA microarray technology [90], [132], [136] is one of a number of technologies that uses the information arising from the genome projects for the exploration of patterns of gene expression on a global scale. Microarray technology makes use of the sequence resources created by the genome projects and other sequencing efforts to answer the questions such as what genes are expressed in a particular cell type of an organism, at a particular time, under particular conditions.

DNA microarray technology exploits the preferential binding of complementary single-stranded nucleic acid sequences. A microarray is typically a glass (or some other material) slide, on to which DNA molecules are attached at fixed locations (spots). There may be tens of thousands of spots on an array, each containing a huge number of identical DNA molecules (or fragments of identical molecules), of lengths from twenty to hundreds of nucleotides. For gene expression studies, each of these molecules ideally should identify one gene or one exon in the genome, however, in practice this is not always so simple and may not even be generally possible due to families of similar genes in a genome. The spots are either printed on the microarrays by a robot, or synthesized by photolithography or by ink-jet printing. There are different ways how microarrays can be used to measure the gene expression levels. One of the most popular microarray applications allows comparing gene expression levels in two different samples. For instance, it allows

comparison of gene expression between normal and diseased (e.g. cancerous) cells (Fig. 2.4).

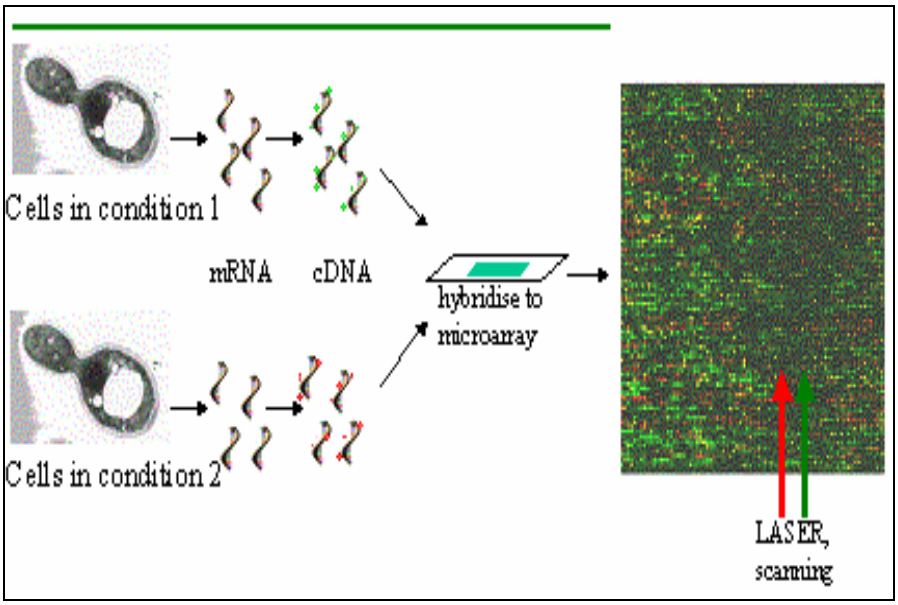

Fig. 2.4 Microarray experiment [62].

The total mRNA from the cells in two different conditions are extracted and labeled with two different fluorescent labels: for example a green dye for cells at condition 1 and a red dye for cells at condition 2 (to be more accurate, the labeling is typically done by synthesizing single stranded DNA that are complementary to the extracted mRNA by an enzyme called reverse transcriptase). Both extracts are washed over the microarray. Labeled gene products from the extracts hybridize to their complementary sequences in the spots due to the preferential binding - complementary single stranded nucleic acid sequences tend to attract each other and the longer the complementary parts, the stronger the attraction. The dyes enable the amount of sample bound to a spot to be measured by the level of fluorescence emitted when it is excited by a laser. For example, if the mRNA from the sample in condition 1 is in abundance, the spot will be green. If the mRNA from the sample in condition 2 is in abundance, it will be red. If both are equal, the spot will be yellow. If neither is present, it will not fluoresce and appear black. Thus, from the

fluorescence intensities and colors for each spot, the relative expression levels of the genes in both samples can be estimated. Since DNA microarray technology only measures mRNA levels rather than protein levels of genes, it should be noted that microarray data alone does not present researchers with a complete picture of the underlying gene expression process. However, although it may be incomplete, gene expression data is still worth exploring as it contains a significant amount of information pertaining to the actual protein levels.

The raw data that are produced from microarray experiments are the hybridized microarray images. To obtain information about gene expression levels, these images should be analyzed. This is called image quantitation. Image quantitation is done by image analysis software. To obtain the final gene expression data matrix from spot quantiations, all the quantities related to some gene have to be combined and the entire matrix has to be scaled to make different arrays comparable (Fig. 2.5).

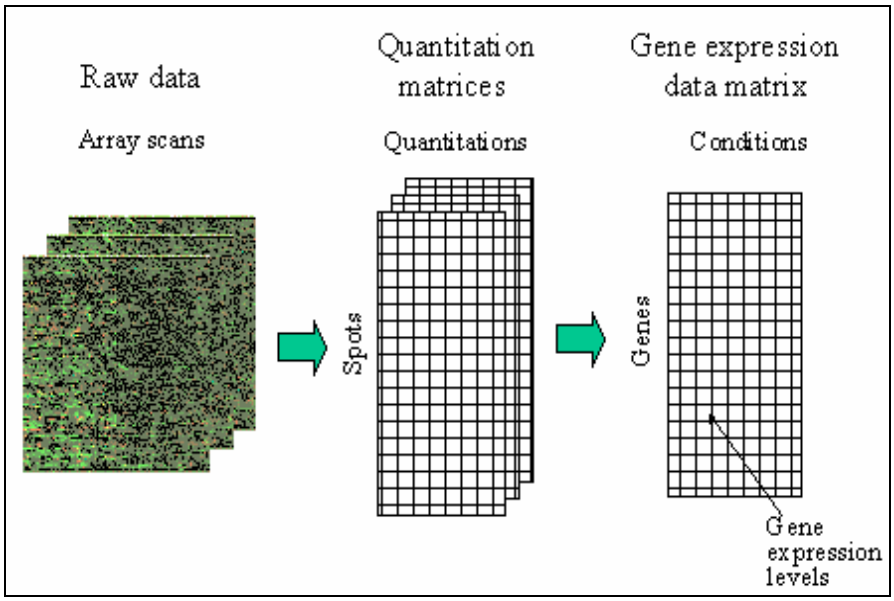

Fig. 2.5 Microarray image analysis [62].

2.2 Knowledge Discovery and Data Mining

Knowledge discovery in databases (KDD) is defined as the nontrivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data [28]. The overall KDD process consists of turning low-level data into high-level knowledge (Fig. 2.6). It is interactive and iterative involving, more or less, the following steps [68]:

1. *Understanding the application domain*: This includes relevant prior knowledge and goals of the application.
2. *Extracting the target dataset*: This is selecting a data set or focusing on a subset of variables.
3. *Data preprocessing and transformation*: This is required to improve the quality of the actual data for mining. This also increases the mining efficiency by reducing the time required for mining the preprocessed data. Data preprocessing involves data cleaning, data transformation, data integration, and data reduction or compression for compact representation.
    a. *Data cleaning*: This consists of some basic operations, such as normalization, noise removal and handling of missing data, reduction of redundancy, etc.
    b. *Data integration*: This operation includes integrating multiple, heterogeneous datasets generated from different sources.
    c. *Data reduction and projection*: This includes finding useful features to represent the data (depending on the goal of the task) and using dimensionality reduction, feature discretization and feature extraction (or transformation) methods.

4. *Data mining*: Data mining constitutes one or more of the following functions, namely, classification and prediction, association analysis, cluster analysis, etc.
5. *Pattern interpretation and evaluation*: This includes interpreting the discovered patterns, as well as the possible visualization of the extracted patterns. Visualization is an important aid that increases understandability from the perspective of humans. One can evaluate the mined patterns automatically or semi-automatically to identify the truly interesting or useful patterns.
6. *Using discovered knowledge*: It includes incorporating the discovered knowledge into the expert system and taking actions based on this knowledge.

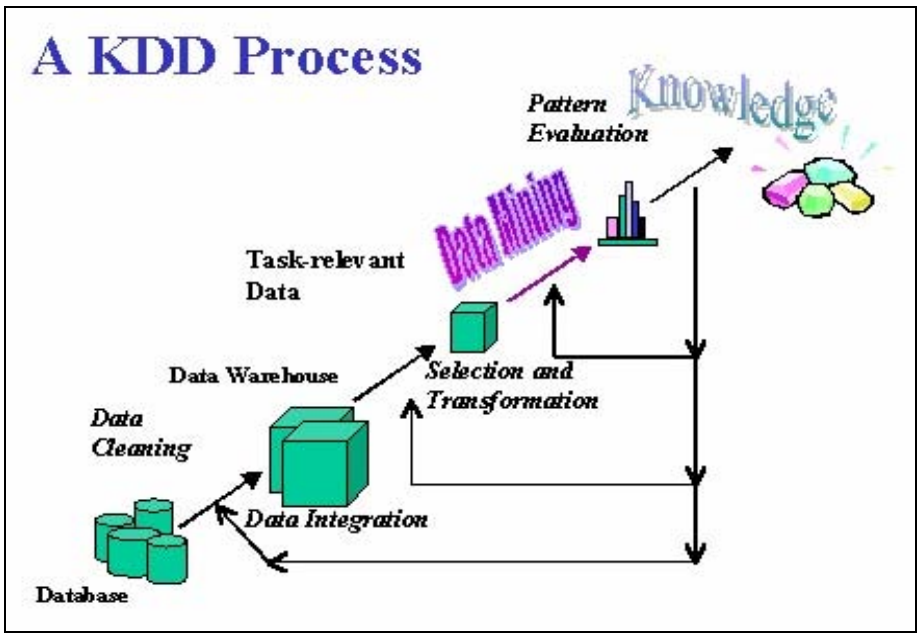

Fig. 2.6 A KDD process [162].

In other words, given huge volumes of heterogeneous data, the objective of the KDD process is to efficiently extract meaningful patterns that can be of interest and useful to the user. The role of interestingness is to filter the large number of discovered patterns and report only those which may be of some use. There are two

approaches to designing a measure of interestingness of a pattern, namely, objective and subjective. The former uses the structure of the pattern and is generally quantitative. The subjective approach, on the other hand, depends additionally on the user who examines the pattern. Two major reasons why a pattern is interesting from the subjective point of view are as follows [138]:

- *Unexpectedness*: When it is surprising to the user and this potentially delivers new information to the user.
- *Actionability*: When the user can act on it to his/her advantage to fulfill the goal.

Data mining is a step in the KDD process consisting of a particular enumeration of patterns over the data. Data mining involves fitting models to or discovering patterns from observed data. The fitted models play the role of inferred knowledge. Deciding whether or not the model reflects useful knowledge is a part of the overall KDD process for which subjective human judgment is usually required. Typically, a data mining algorithm constitutes some combination of the following three components:

- *The model*: The function of the model (i.e., classification, clustering, etc.) and its representational form (i.e., linear discriminants, decision trees, etc.).
- *The preference criterion*: A basis for preference of one model or set of parameters over another, depending on the given data. The criterion is usually some form of goodness-of-fit function of the model to the data.
- *The search algorithm*: The specification of an algorithm for finding particular patterns, given the data, model and a preference criterion.

A particular data mining algorithm is usually an instantiation of the model-preference-search components. Some of the common model functions in current data mining practice include [68]:

1. *Association analysis*: This function mines or generates rules from the data. Association rule mining refers to discovering associations among different attributes (for details, refer to Section 2.2.1).
2. *Classification*: This model function classifies a data item into one of several predefined categorical classes (for details, refer to Section 2.2.2).
3. *Cluster analysis*: This function maps a data item into one of several clusters, where clusters are natural groupings of data items based on similarity metrics or probability density models (for details, refer to Section 2.2.3).
4. *Prediction*: The purpose of this model function is to map a data item to a real-valued prediction variable.
5. *Summarization or condensation*: This function provides a compact description for a subset of data. Data compression play a significant role for multimedia data, because of the advantage it offers to compactly represent the data with a reduced number of bits, thereby increasing the database storage bandwidth.
6. *Sequence analysis*: This models sequential patterns, like gene and protein sequences. The goal is to model the states of the process generating the sequence.

2.2.1 Association Analysis

Association analysis is the discovery of association rules showing attribute-value conditions that occur frequently together in a given set of data, for example, a set of transactions. A typical example of association analysis is the market basket analysis (Fig. 2.7). This process analyzes customer buying habits by finding associations between the different items that customers place in their shopping baskets. The discovery of such associations can help retailers developing marketing strategies by gaining insight into which items are frequently purchased together by customers. For example, given customers are buying milk, how likely are they also buying bread on the same trip to the supermarket can be revealed. Such information can lead to increase sales by helping retailers perform selective marketing and plan their shelf places. For example, placing milk and bread within close proximity may further encourage the sale of these items together within single visit to the store.

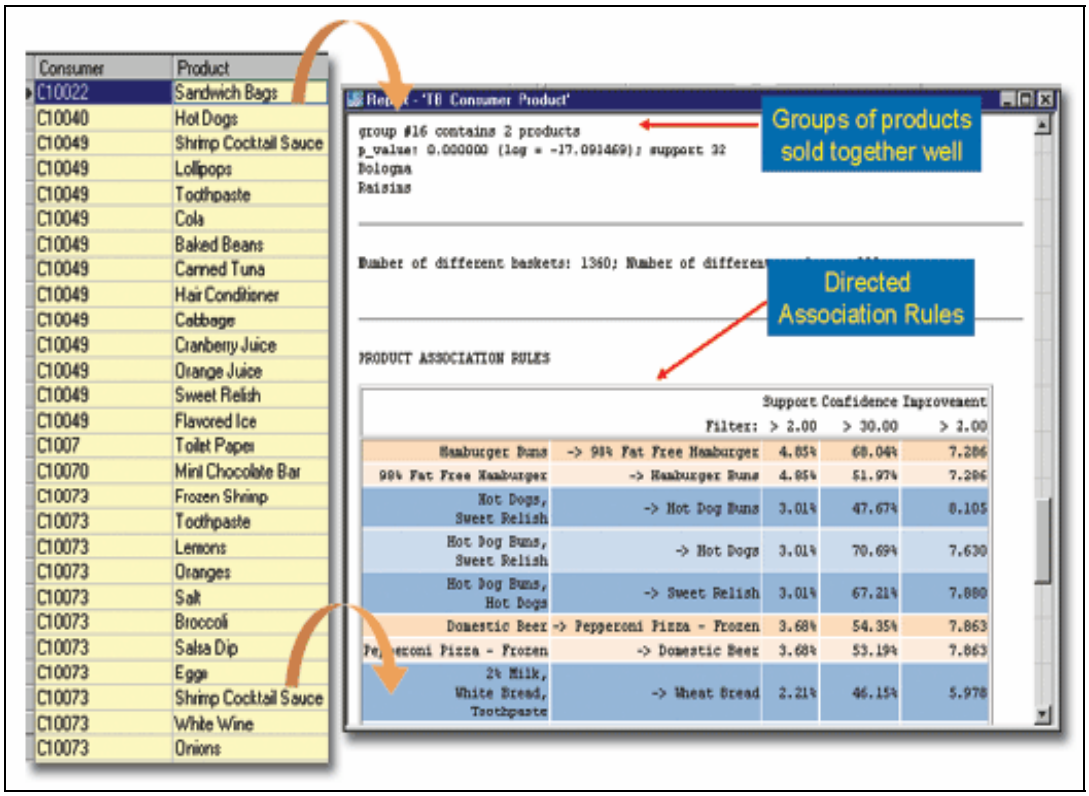

Fig. 2.7 Market basket analysis [66].

Much work has been done (pioneered by [1]-[2]) to find associations among items in large groups of transactions. Typically, this consists of two steps as follows (additional filtering steps or interestingness measures can be applied, if applicable):

1. *Find all frequent itemsets*: Each of these itemsets must occur at least as frequently as a pre-determined minimum support count (a set of items is referred to as an itemset [68]).
2. *Generate association rules from the frequent itemsets*: These rules must satisfy minimum support and minimum confidence.

A rule is normally expressed in the form $X \Rightarrow Y$, where X and Y are sets of attributes of the data set, and the implication holds with the $support \geq s$ and $confidence \geq c$, where s and c are user-defined thresholds. This implies that transactions which contain X also contain Y. For example,

$$IF < some\_conditions\_satisfied >$$

$$THEN < predict\_values\_for\_some\_other\_conditions >$$

$$[support \geq s, confidence \geq c].$$

A sample rule could be of the form as follows:

$$IF\ (salary \geq 20000)\ \ AND\ (unpaid\_loan = "no")$$

$$THEN\ (select\_for\_loan = "yes")$$

$$[support \geq 20\%, confidence \geq 75\%].$$

2.2.2 Classification

Classification is also described as supervised learning [147]. Let there be a database of records, each assigned a class label. The objective is to develop a model for each

class. An example of a model with good credit is $25 \le age \le 40$ and $income > 40K$ or $married = "yes"$. Sample applications of classification include:

- Signature identification in banking or sensitive document handling (match or no match).
- Digital fingerprint identification in security applications (match or no match).
- Credit card approval depending on customer background and financial credibility (good or bad).
- Bank location considering customer quality and business possibilities (good or fair or poor).
- Treatment effectiveness of a drug in the presence of a set of disease symptoms (good or fair or poor).

The input to the classification algorithm is, typically, a set of training records with several attributes (Fig. 2.8). There is one distinguished attribute called the dependent attribute. The remaining predictor attributes can be numerical or categorical in nature. A numerical attribute has continuous, quantitative values. A categorical attribute, on the other hand, takes up discrete, symbolic values. If the dependent attribute is categorical, the problem is called classification with this attribute being termed the class label. However, if the dependent attribute is numerical, the problem is termed prediction/regression. The goal of classification is to build a concise model of the distribution of the dependent attribute in terms of the predictor attributes. The resulting model is used to assign values to the testing records, where the values of the predictor attributes are known but the dependent attribute is to be determined (Fig. 2.9).

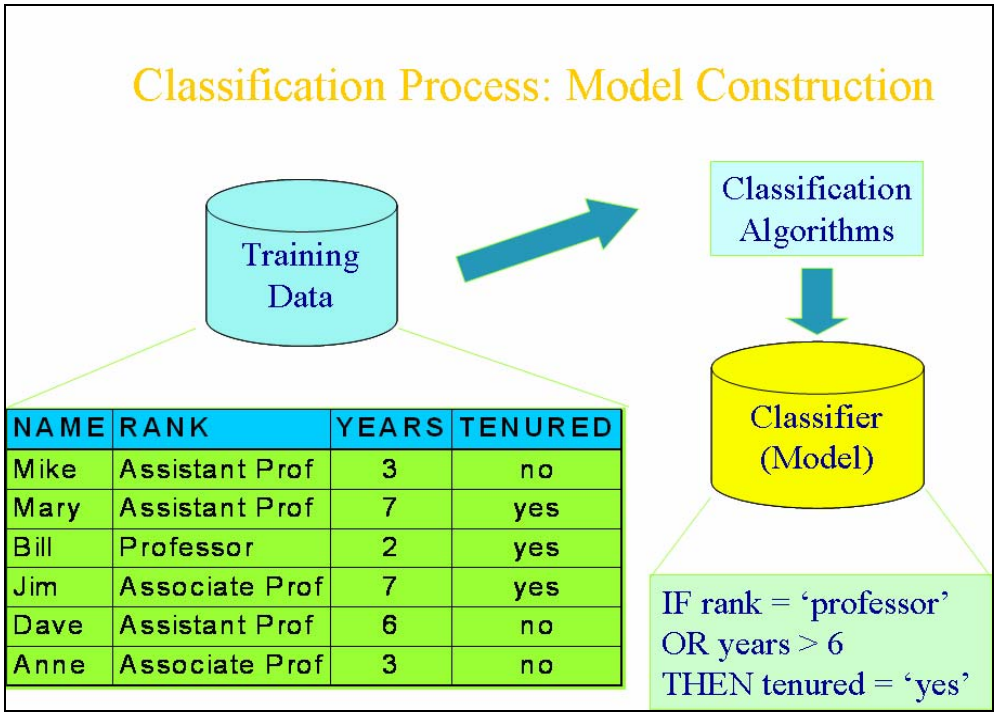

Fig. 2.8 Classification process (training process) [43].

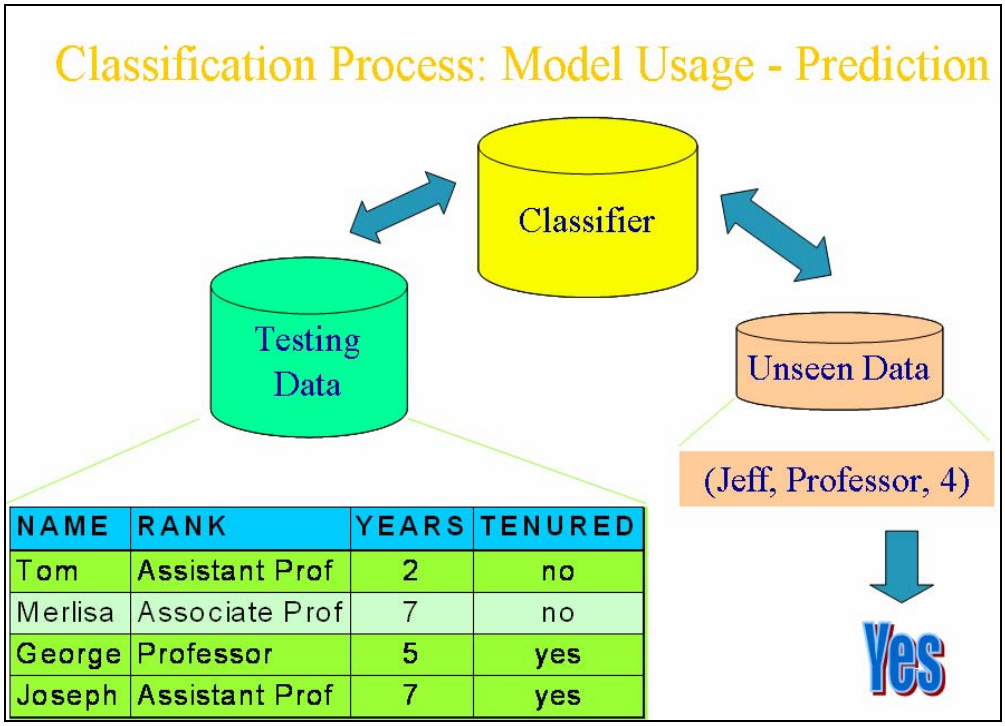

Fig. 2.9 Classification process (testing process) [43].

Classification methods can be briefly categorized as follows:

1. Decision trees [129], [131], which divide a decision space into piecewise constant regions. Typically, an information theoretic measure is used for assessing the discriminatory power of the attributes at each level of the tree.

2. Probabilistic models, which calculate probabilities for hypotheses based on Bayes' theorem [147].
3. Nearest-neighbor classifiers, which compute minimum distance from instances [147].
4. Neural networks [99], which partition by nonlinear boundaries. These incorporate learning, in a data-rich environment, such that all information is encoded in a distributed fashion among the connection weights.

2.2.3 Cluster Analysis

A cluster is a collection of data objects which are similar to one another within the same cluster but dissimilar to the others in other clusters. Cluster analysis is concerned with the problem of grouping a set of data objects into clusters (Fig. 2.10). Clustering is also called unsupervised classification, where no predefined classes are assigned [147].

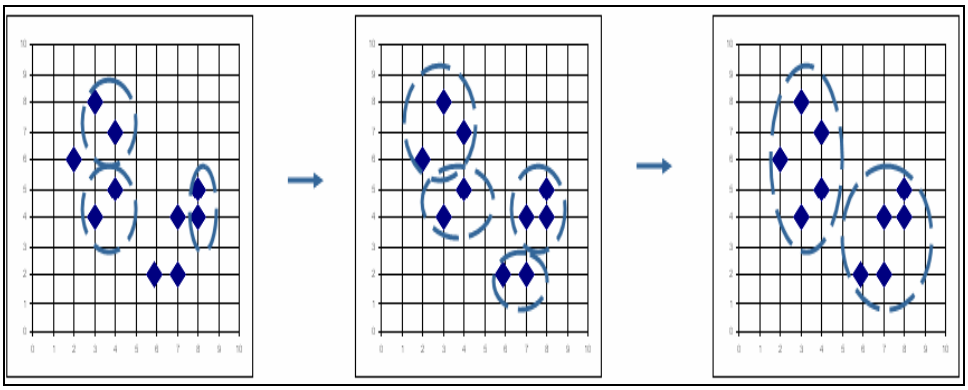

Fig. 2.10 Clustering process [43].

Some example applications of clustering include:

- *Spatial data analysis*: Creating maps in geographic information systems by clustering feature spaces and detecting spatial clusters [68].

- *Multimedia computing*: Finding the cluster of images containing objects of similar color and shape from a multimedia database [101].
- *Bioinformatics*: Discovering clusters of co-expressed genes in gene expression data [11].
- *Biometrics*: Creating clusters of facial images with similar fiduciary points [101].
- *WWW*: Clustering web-log data to discover groups of similar access patterns [68].

A good clustering algorithm will produce high-quality clusters with high intra-class similarity and low inter-class similarity, and the quality of a clustering result depends on the similarity measure used by the algorithm. Some popular clustering algorithms and the similarity measures will be discussed in details in Section 2.3.3.

Clustering approaches can be broadly categorized as:

1. *Partitional*: Create an initial partition and then use an iterative control strategy to optimize an objective.
2. *Hierarchical*: Create a hierarchical decomposition (dendogram) of the set of objects using some termination criterion.
3. *Density-based*: Use connectivity and density functions.
4. *Grid-based*: Create multiple-level granular structure, by quantizing the feature space in terms of finite cells.

2.3 Data Mining Techniques for Bioinformatics: A Survey of Related Work

Recently, there have been many data mining and statistical techniques [68], [70], [99] available for use in biological data analysis. In Section 2.3.1, we briefly review some existing tools and databases developed specifically for solving bioinformatics problems [10]-[11], [51], [91], [98], [142], [155]. In Sections 2.3.2 and 2.3.3, the related work of this thesis will be given.

2.3.1 Overview of Bioinformatics Problems and Tools

Sequencing of a complete genome (i.e., yeast, human, etc.) and subsequent annotation of the features (i.e., genes, promoter regions, etc.) in the genome are the two important problems of genome analysis [11]. The first problem is related to sequence assembly. To solve this problem, there are many tools have been developed such as CAP3 [72]. The second problem is related to the prediction of genes in the genome. Since, the eukaryotic gene structure is much more complex due to the intron/exon structure, gene prediction is therefore easier and more accurate in prokaryotic than eukaryotic genomes. Several software tools [12]-[13], [140], such as GeneMark, GeneScan, Glimmer, and GRAIL can accurately predict genes in prokaryotic and eukaryotic genomes respectively.

Over the past decade, many techniques have been used for DNA and protein sequence analysis such as sequence alignment and motif finding [98]. For biological sequence alignment, many existing tools were based on a dynamic programming algorithm [139], including pair-wise alignment tools such as BLAST [3] and multiple sequence alignment tools such as ClustalW [71]. Hidden Markov model

(HMM) [133] is another widely used method. HMMER [44], which is used to find conserved sequence domains in a set of related protein sequences, is one of the popular HMM tools. Other interesting problems include promoter and protein functional motif finding. Several probability models and stochastic methods have been applied to these problems, including expectation maximization algorithms and Gibbs sampling methods [10].

Common problems of biological structure analysis include classification and prediction, comparison, and visualization of the molecular structures (i.e., RNA secondary structure, protein secondary and tertiary structures, etc.) [14]. For these problems, some popular software tools include Mfold [14] for RNA secondary structure prediction, DALI [81] for structural alignment, Cn3d and Rasmol [82] for viewing the 3D structures. Protein structure databases, such as PDB [25], SCOP [110], and CATH [125], and the associated tools also play an important role in biological structure analysis [14].

Biological processes in a cell form different biological pathways among genes and their functional products. Biological pathway analysis is concerned with the problems of modeling and visualizing these pathways [15]. Several tools and databases have been developed and are commonly used, including GenMapp [15], KEGG database [88] and MetaCyc [89].

With the advent of DNA microarray technology and the genome sequences of many model organisms, the simultaneous monitoring of expression levels for all genes in a genome has become possible. Although a relatively new technology, the use of microarrays has spread to almost all branches of biochemistry and molecular biology, for example, in drug discovery [67]. Applications of DNA microarray technology have resulted in generating very many gene expression databases [19],

[91], [143]. Effective data mining techniques are therefore needed to analyze such data. Hierarchical clustering [52] was the first clustering method applied to the problem of finding clusters of co-expressed genes in the expression data. Since then many popular clustering methods have been used [130], such as *k*-means and self-organizing map. By combining sequence analysis methods, one can also identify common regulatory motifs from the co-expressed genes discovered by the clustering method. Furthermore, any correlations among gene expression profiles can be modeled by advanced techniques and can hopefully help to reverse-engineer the genetic networks in a cell. With sophisticated gene expression data analysis tasks, there is much room for research and development of advanced, effective and scalable data mining techniques. In the following sections, two important bioinformatics problems and also the related work of this thesis will be presented respectively.

2.3.2 Reconstructing Gene Regulatory Networks from Gene Expression Data

Large-scale monitoring of gene expression such as DNA microarray [90], [132], [136] is considered to be one of the most promising techniques for making the reconstruction of gene regulatory networks (GRNs) [152] feasible. A GRN [15] is a complex biological system in which genes interact with each other indirectly via the proteins they create to perform various cellular processes. Better understanding of gene interactions may therefore lead to better understanding of how cellular processes are carried out to accommodate changes in the external environment [157]. Unfortunately, since living cells contain thousands of genes with each interacting

with one or more other genes, the task of inferring the structures of GRNs is very difficult.

In an attempt to do so, some approaches have been developed. They include biochemically driven approaches [31], [94], [122], [150], Boolean network approaches [4]-[5], [134], Bayesian network approaches [75], [126], [164], and data mining approaches [17], [32], [52], [102], [141]-[142]. They are described briefly in the following.

Biochemically inspired models [122] are based on the reaction kinetics [6] between different components of a GRN. Reaction kinetics provides a framework by which biochemical reactions between molecular compounds can be described. For example, if a transcription factor, produced by a gene, say g_j, is brought together with the DNA sequence it is selective to, say, in the regulatory region of g_i, it might react with a rate k_1 to form a compound with the DNA, and then also dissociate with a rate k_2 from the DNA. This process is started by a high expression rate of g_j, followed by diffusion towards the binding site of g_i, and then it is decreased by spatial diffusion away from g_i and other chemical reactions. Reconstructing GRNs based on biochemically inspired models has the advantage that these models can be most directly related to biological processes. Unfortunately, they also have some disadvantages. For example, most of the biochemically relevant reactions under participation of proteins do not follow linear reaction kinetics. Many proteins undergo conformational changes after reactions and these also change their chemical behavior. In particular, in many regulatory DNA regions transcription factor binding can show cooperative or competitive effects, which are nonlinear and mostly unknown. Moreover, the full network of regulatory reactions is very complex and

hard to disentangle in a single step. One needs to know the kinetic equations of all the different interactions to do so. Unfortunately, the types of reactions and their parameters are often unknown and at present, the data collected are not sufficient for regulatory networks to be understood at this level of details. As a result, some researchers used approximations to reaction-kinetic formulations to arrive at systems of coupled differential equations [31], [94], [150] for describing the time course of gene expression levels. However, the primary disadvantage of these approaches is that they are computationally intensive.

In a Boolean network [4]-[5], [134], the expression state of a gene is represented by a Boolean variable (ON or OFF) and interactions between genes are represented by Boolean functions. The Boolean functions determine the expression state of a gene on the basis of the expression states of some other genes. The Boolean network approaches require that a number of assumptions be made to simplify analysis. For example, the activation of a single gene is represented as a Boolean switch that can either be on or off, and regulatory control of a gene is describable by a combination of Boolean logic rules, such as AND, OR and NOT. As there is little knowledge of connectivity patterns in real biological networks, for example, an ensemble approach has been used to generate large numbers of randomly connected networks with randomly chosen Boolean updating functions. The goal was to measure the generic properties of certain classes of networks and observe how the global dynamics was affected by these local interactions. This model of Boolean network has two primary parameters: (i) network size, N, which specifies the number of elements in the network, and (ii) K, the number of inputs regulating the activity of each element. Each of the N elements is associated with a rule table specifying outputs for each of the 2^K possible input combinations. The rule tables for each element can also be

defined in a number of different ways. If the several assumptions as discussed above can be validly made, then the use of Boolean networks has the advantage that the computational requirements of simulating regulatory systems are massively reduced, allowing the exploration of much larger systems. Unfortunately, the validity of the above assumptions has been questioned by a number of researchers, particularly among those in the biological community, where there is a perceived lack of connection between simulation results and empirically testable hypotheses [53].

Bayesian network [75], [126], [164] is a probabilistic model that describes the multivariate probability distribution of a set of variables, where each variable only depends on its parents. The basic idea is to display the associations among the variables, namely the conditional dependencies and independencies, by means of a directed acyclic graph. In the context of GRN, each node of a Bayesian network can be considered as representing a gene, and each edge between connecting genes hints towards an interaction between them. If this edge is directed, it can, under certain assumptions, be interpreted as a causal relationship and can be inferred as one gene controlling another. To construct Bayesian networks, one can use Bayesian statistics [33] to find the network structures and the corresponding model parameters that best describe the probability distribution for which the data is drawn. The goodness-of-fit of a network with respect to the data can be assessed by assigning a score based on a statistically motivated scoring function such as the Bayesian score [76]. Unfortunately, this learning task is NP-hard, especially for high-dimensional data, such as microarray data which often contain thousands of expressed genes. To make this feasible, the size of the search space must be restricted by such heuristics as the construction of small sub-networks or in a slightly more sophisticated way, by restricting the maximum number of parents of each variable. Another problem with

the Bayesian network approaches is the effect of small sample size and this can make the estimation of the many parameters required for a Bayesian network difficult, if not impossible. As shown in [46], a Bayesian network constructed from small amounts of gene expression data is most likely not able to detect all gene interactions that are supposed to be present in a network.

Other than the above approaches, some recent attempts have been made to infer the structures of GRNs using data mining approaches. Given a set of data, the goal of data mining [68] is to discover hidden regularities and structures in it. As opposed to the above hypothesis-driven approaches, which search for known, pre-defined patterns in a set of data, data mining approaches are data-driven. Instead of requiring patterns to be known ahead of time, they search automatically for patterns that are hidden in the data. For GRNs reconstruction, several data mining techniques have been proposed [17], [32], [52], [102], [141]-[142].

Given a gene expression profile characterized by a set of different experimental conditions, clustering algorithms have been used to group co-expressed genes into clusters according to how similar they are to each other. Co-expressed genes are genes that have similar transcriptional responses to the external environment (i.e., temperature, pH value, pressure, etc). This can be an indication that they might be co-regulated by the same regulatory mechanism. Among all clustering algorithms, hierarchical agglomerative clustering has been more popularly used with gene expression data [17], [52], [141]. It performs its tasks by a series of successive fusion of genes into clusters. This fusion process is guided by a measure of similarity between clusters so that clusters that are similar to each other are merged. This fusion process is repeated until all clusters are merged into a single cluster. The result of the fusion process is normally presented in the form of a two-dimensional

hierarchical structure, called a dendrogram. The genes falling along each branch in a dendrogram form a cluster. Depending on user-preferences, a specific number of clusters can be obtained from the dendrogram by cutting across the branches at a specific level. The key step in performing hierarchical clustering is the measure of similarity between two genes, and for this, the correlation coefficient [16] is often used. Since this approach can only determine if two genes have a significant linear relationship with each other, the regulatory relationships, such as which gene affects which other genes, cannot be discovered. In addition to this, the discovered relationships cannot be explicitly revealed for possible interpretation.

Artificial neural networks [99] are mathematical models of information processing originally inspired by networks of neurons in the brain. A neural network typically consists of a collection of nodes, some of which may be designated as input or output nodes, connected by weighted links. A transfer function is associated with each node and this transfer function transforms a set of weighted input signals into an output signal. Neural networks can be trained to match particular patterns of activation via some learning processes. Mathematically, it is possible to create a mapping between a neural network and a GRN. Conceptually, a relatively straightforward analogy may be drawn between a neural network, in which the constituent elements are neurons and the links are synaptic interactions, and a GRN, in which the elements are genes and the links are regulatory interactions. Based on such a conceptual model, there are some successes using neural networks to model GRNs [142]. However, since the training of neural networks requires many free parameters, it should be noted that the trials-and-errors required determining the best set of parameters are usually very time consuming. In fact, even with the most straightforward neural network formulation, it also requires weight parameters in the

order of the square of the number of genes. In addition, there are other free parameters describing the nonlinear functions. Given that the number of microarray measurements/samples collected usually ranges between tens to a few hundreds, it is very difficult for these parameters to be accurately estimated from the data in practice. In addition to the requirement for estimating many parameters, it should also be noted that the neural network approach is a black-box approach. The patterns it discovers cannot be explicitly revealed for interpretation.

To use the decision-tree-based approaches [32], [102], the expression values of a gene, say g_i, is first divided into finite number of states. Whether the state of g_i can be determined by the states of other genes is then decided by checking whether these genes can predict the state of g_i. If the prediction accuracy is high, then we can conclude that the state of g_i can be determined by the states of other genes. Of the existing approaches [68] that can be used for such prediction, C4.5 [131] is the most popular one. C4.5 uses a greedy procedure to select the attributes that yield the maximum information gain in order to recursively partition the training set. The leaf nodes of the tree correspond to the states/classes of the gene. And the discovered patterns are represented in the form of decision trees. The advantages of this approach against others, which have been discussed above, are that it can identify genes affecting a target gene in an explicit manner, and also the discovered patterns can be used to predict how a gene would be affected by other genes from the unseen tissue samples. However, due to the fact that the pruning methods which the decision-tree-based classifiers adopt are based on hill-climbing approaches, important information can be overlooked [131]. In addition, these classifiers still have to overcome some other problems to effectively deal with imperfect data in classification tasks [34].

The biochemically driven approaches, Boolean network approaches, Bayesian network approaches, and data mining approaches are the most common techniques for GRNs reconstruction. Recently, the use of evolutionary computation to reconstruct underlying GRNs from gene expression data is also a growing research area and is getting a lot of attention from bioinformatics community [7], [37], [165]. However, besides their own limitations, many approaches can only be used to generate hypotheses about the presence or absence of interactions between genes so that laboratory tests can be carried out later for verification. It should be noted that many of them are not intended to be used to predict, for example, how a gene would be affected by other genes from the unseen samples (i.e., expression data that is not in the original database). This can make statistical verification of the reliability of the discovered patterns (relationships between genes) difficult.

2.3.3 Clustering of Gene Expression Data

Gene expression, as discussed before, is the process by which a gene's coded information is converted into the structures present and operating in a cell. Gene expression occurs in two major stages: transcription and translation. During transcription, a gene is copied to produce an mRNA molecule (a primary transcript) with essentially the same sequence as the gene; and during translation, proteins are synthesized based on the mRNA molecule. If one would like to prevent undesirable genes, such as cancerous genes, from expressing, the transcription process should be prevented as much as possible from taking place so that the corresponding undesirable functional proteins will not be synthesized [27], [73].

To prevent the transcription process of undesirable genes from taking place, a set of transcription factor binding sites must be located. These sites consist of untranscribed nucleotide sequences located on the promoter regions (non-coding regions) of the genes and are responsible for activating and regulating the process. If they are located, we can then bind appropriate molecules, such as protein repressors, to these sites so that the genes they correspond to cannot be activated [27]. To locate these transcription factor binding sites, co-expressed genes may need to be identified. Co-expressed genes are genes that have similar transcriptional responses to the external environment. This can be an indication that they might be co-regulated by the same transcription factors and therefore might have common binding sites. To identify co-expressed genes, one can cluster gene expression data obtained by performing microarray experiments [29]-[30], [45], [57], [92]-[93], [141]. Genes that are grouped into a cluster are likely to be co-expressed genes. By analyzing the promoter regions of these genes, we may be able to discover patterns (motifs), which have relatively high occurring frequencies compared to other sequence fragments that are possible binding sites of these genes [163].

Given a database of records each characterized by a set of attributes, the clustering problem is concerned with discovering interesting groupings of records based on the values of the attributes. Many clustering algorithms [68] have been developed to tackle different clustering problems in a variety of application domains and they have been proven to be very effective. Recently, some of them, including the hierarchical agglomerative clustering algorithm [156], the *k*-means algorithm [100] and the Self-Organizing Map (SOM) [85], have been used to cluster gene expression data [52], [148]-[149] respectively. These algorithms are described briefly below.

To discover clusters in the data, the hierarchical agglomerative clustering algorithm [52], [156] performs a series of successive fusions of records into clusters. The fusion process is guided by a measure of similarity between clusters so that clusters that are similar to each other are merged. This fusion process is repeated until all clusters are merged into a single cluster. The results of the fusion process are normally presented in the form of a two-dimensional hierarchical structure, called a dendrogram. The records falling along each branch in a dendrogram form a cluster. Depending on user-preferences, a specific number of clusters can be obtained from the dendrogram by cutting across the branches at a specific level.

Comparing to the hierarchical agglomerative clustering algorithm that does not require users to specify the number of clusters ahead of time, users of the *k*-means algorithm [100], [148] are required to do so. Given a data set, the *k*-means algorithm can group the records into *k* clusters by initially selecting *k* records as centroids. Each record is then assigned to the cluster associated with its closest centroid. The centroid for each cluster is then re-calculated as the mean of all records belonging to the cluster. This process of assigning records to the nearest clusters and re-calculating the position of the centroids is then performed iteratively until the positions of the centroids remain unchanged.

The Self-Organizing Map (SOM) algorithm [85], [149] is one of the best-known artificial neural network algorithms. It can be considered as defining a mapping from *M*-dimensional input data space onto a map (a regular two-dimensional array of neurons) so that every neuron of the map is associated with an *M*-dimensional reference vector. The reference vectors together form a codebook. The neurons of the map are connected to adjacent neurons by a neighborhood relation, which dictates the topology of the map. In the basic SOM algorithm, the topology and the

number of neurons remain fixed from the beginning. The number of neurons determines the granularity of the mapping, which has an effect on the accuracy and generalization of the SOM. During the training phase, the SOM forms an elastic net that folds onto the cloud formed by input data. The algorithm controls the net so that it strives to approximate the density of the data. The reference vectors in the codebook drift to the areas where the density of the input data is high. Eventually, only few codebook vectors lie in areas where the input data is sparse. After the training is over, the map should be topologically ordered. This means that *n* topologically close (based on, say, the Euclidean distance or the Pearson correlation coefficient) input data vectors map to *n* adjacent map neurons or even to the same single neuron. With this idea, SOM has been used successfully in various application areas.

Despite some successes with existing clustering algorithms in gene expression data analysis [17], [52], [74], [130], [148]-[149], there is still no single clustering algorithm that is the most dominant gene expression data clustering algorithm. This may be a result of their use of such metrics and functions as the Euclidian distance measure or the Pearson correlation coefficient [84] that do not differentiate between the importance of different variables when measuring similarities. They also do not give very accurate measurements when the data concerned are very noisy. As these metrics and functions measure only pair-wise distances, the measurements obtained could be too local. Clustering algorithms based only on the local pair-wise information may, therefore, miss important global information. In addition to these deficiencies, clustering results obtained with the use of many clustering algorithms could be difficult to interpret. For example, although one can visualize the result of hierarchical clustering as a tree-like dendrogram and note correlations between

genes, it is the users' responsibilities to discover the similarities and differences between various clusters and to decide on the number of clusters and the cluster boundaries to form. To do so, users need to have prior knowledge about the data. Similarly, for the *k*-means algorithm and SOM, users have to decide on the number of clusters to partition a data set into. They also have to use a separate technique to uncover underlying patterns in the clusters.

Chapter 3

Reconstructing Gene Regulatory Networks from Gene Expression Data

Recent developments in large-scale monitoring of gene expression, such as DNA microarray technology, have made the reconstruction of gene regulatory networks (GRNs) feasible. Before one can infer the structures of GRNs, it is important to identify, for each gene in a network, which other genes can affect its expression and how they can affect it. Many existing approaches such as the biochemically driven approaches, Boolean network approaches, Bayesian network approaches, and data mining approaches (as discussed in Chapter 2, Section 2.3.2) have been applied to GRNs reconstruction. Besides their own limitations, many of these approaches can only be used to generate hypotheses about the presence or absence of interactions between genes so that laboratory tests can be carried out later for verification. Since many of them are not intended to be used to predict, for example, how a gene would be affected by other genes from the unseen samples (i.e., expression data that is not in the original database), this makes statistical verification of the reliability of the discovered relationships/patterns difficult. To better infer the structures of GRNs, we propose to use an effective data mining technique in this chapter.

The proposed association-discovery technique [113]-[115] is able to discover interesting association relationships between genes and can handle very noisy high-dimensional time-dependent gene expression data. By computing an average gene expression value which serves as a reference point for how large the value is, this

technique can discover interesting sequential associations between genes such as “if a gene is highly expressed, its dependent gene is then lowly expressed in the next time point”, etc. These findings can not only allow hidden regularities to be easily interpreted, they can also determine if a gene is supposed to be activated or inhibited and can be used to predict how a gene would be affected by other genes from the unseen samples. In Section 3.1, the proposed association-discovery technique, which can be used to discover interesting association relationships between genes, will be introduced. Given time-dependent gene expression data, this technique can also be used to discover interesting sequential associations, the details of the sequential association discovery approach will be given in Section 3.2.

3.1 An Association-Discovery Technique

To describe the proposed association-discovery technique [113]-[115], let us assume that we are given a set of gene expression data, *G*, consisting of the data collected from *M* genes in *N* experiments carried out under different experimental conditions. Let us represent the data set as a set of *M* genes, $\mathrm{G} = \{g_1, \ldots, g_j, \ldots, g_M\}$, with each gene, g_j , $j = 1,...,M$, characterized by *N* different experimental conditions, $E_1, \ldots, E_i, \ldots, E_N$, whose values, $e_{1j}, \ldots, e_{ij}, \ldots, e_{Nj}$, where e_{ij} represents the expression value of the j^{th} gene under the i^{th} experimental condition.

Since gene expression values can be described in a finite number of different states such as “expressed” and “not expressed”, “upregulated” and “downregulated”, or other different number of states, etc., we define two different states for it: “highly expressed (H)” and “lowly expressed (L)” [32]. These states are defined below.

Let $\bar{e}_j = \frac{\sum_{i=1}^{N_j} e_{ij}}{N_j}$ be the average expression value of g_j under N_j different experimental conditions, from E_1 to E_{N_j}, where N_j is the total number of experimental conditions in which the expression of g_j is recorded and $N_j \leq N$ due to possible missing values in the expression profile of g_j. Given $\bar{e}_j$, the expression data obtained for g_j in each of the experimental conditions can then be mapped to H if $e_{ij} > \bar{e}_j$ and L if $e_{ij} \leq \bar{e}_j$, where $i = 1,..., N_j$. With such mapping, instead of e_{ij}, g_j can now be considered as taking on a transformed expression value, $s_{ij} \in \{H, L\}$ under the experimental condition E_i. This process of transformation is performed for each of the M genes in G (Fig. 3.1).

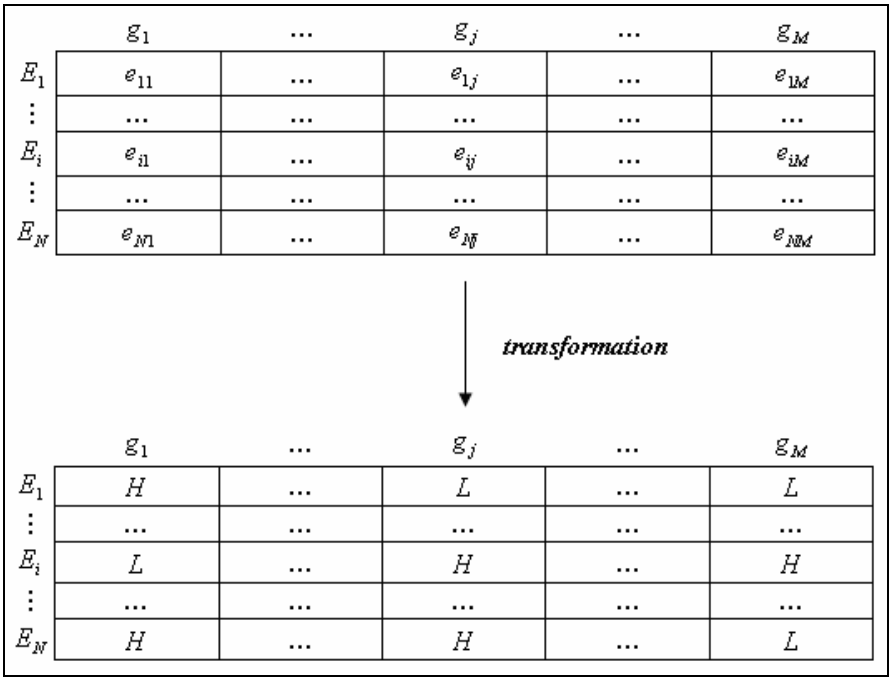

Fig. 3.1 The transformation process.

After transformation, the association-discovery technique is used to discover interesting association relationships between genes. This technique consists of two steps as follows.

Step 1 - Discovering interesting associations between the states of genes:

Interesting association relationships are discovered by detecting for associations between the states of the target gene, g_q, and the states of each other gene, g_j, in a set of training samples, where $q \neq j$ and $g_q \in G$. To do so, we let obs_{pk} be the observed total number of experimental conditions, $E_1,...,E_l$ where $l \leq N'$, that the state of g_q is $s_q^{(p)}$ and the state of g_j is $s_j^{(k)}$, where $s_{1q} = ... = s_{lq} = s_q^{(p)}$, $s_{1j} = ... = s_{lj} = s_j^{(k)}$, $s_q^{(p)}, s_j^{(k)} \in \{H, L\}$, $p,k = 1,...,|\{H,L\}|$, and $N' = \sum_{p,k} obs_{pk} \leq N$ due to possible missing values in the data. We also let $\exp_{pk} = \frac{obs_{p+} obs_{+k}}{N'}$ be the expected total under the assumption that the state, $s_q^{(p)}$, of g_q is independent of whether the state of g_j is $s_j^{(k)}$, where $obs_{p+} = \sum_{k=1}^{|\{H,L\}|} obs_{pk}$, and $obs_{+k} = \sum_{p=1}^{|\{H,L\}|} obs_{pk}$. Given obs_{pk} and $\exp_{pk}$, we are interested in determining whether obs_{pk} is significantly different from $\exp_{pk}$ (Fig. 3.2).

obs_{pk} $\exp_{pk}$ d_{pk}	H	L
H	5 2.16 2.45	4 2.64 1.09
L	1 2.88 −1.48	2 3.52 −1.11

Fig. 3.2 Frequency table for the states of the target gene (g_q , column) and the states of other gene (g_j , row).

To determine if this is the case, we can use the standardized residual [77] to scale the difference as follows:

$$z_{pk} = \frac{obs_{pk} - \exp_{pk}}{\sqrt{\exp_{pk}}}. \tag{3.1}$$

This statistic approximates the standard normal distribution only when the asymptotic variance of z_{pk} is close to one. Therefore, it is, in practice, adjusted by its variance for a more precise analysis. The new test statistic, which is called the adjusted residual, can be expressed as follows:

$$d_{pk} = \frac{(obs_{pk} - \exp_{pk}) \Big/ \sqrt{\exp_{pk}}}{\sqrt{v_{pk}}} = \frac{z_{pk}}{\sqrt{v_{pk}}}, \tag{3.2}$$

where v_{pk} is the maximum likelihood estimate of its asymptotic variance and is defined as:

$$v_{pk} = (1 - \frac{obs_{p+}}{N'})(1 - \frac{obs_{+k}}{N'}). \tag{3.3}$$

This statistic has an approximate standard normal distribution [8], [34], [36], [158] and an association is considered to be interesting if the test statistic is statistically significant. In other words, if $d_{pk} > 1.96$ (Eq.(3.2)), we can conclude, with a

confidence level of 95 percent, that the state, $s_j^{(k)}$, of g_j is significantly associated with the state, $s_q^{(p)}$, of the target gene, g_q (Fig. 3.2).

Step 2 – Determining the weight of evidence of the associations:

Using Eq.(3.2), we can determine whether $s_j^{(k)}$ is significantly associated with $s_q^{(p)}$. If it is the case, then it can be utilized to construct a characteristic description of $s_q^{(p)}$. Such an association is not completely deterministic and the uncertainty associated with it is quantified using a measure defined so that if the state of g_j is $s_j^{(k)}$, then it is with certainty W(State = $s_q^{(p)}$ / State ≠ $s_q^{(p)}$ | $s_j^{(k)}$) that the state of g_q is $s_q^{(p)}$, where W, the weight of evidence measure [124], is defined in terms of the mutual information, $I(s_q^{(p)} : s_j^{(k)})$, between $s_q^{(p)}$ and $s_j^{(k)}$ as follows:

$$I(s_q^{(p)} : s_j^{(k)}) = \log \frac{\Pr(s_q^{(p)} \mid s_j^{(k)})}{\Pr(s_q^{(p)})}. \qquad (3.4)$$

$I(s_q^{(p)} : s_j^{(k)})$ is positive if $\Pr(s_q^{(p)} \mid s_j^{(k)}) > \Pr(s_q^{(p)})$. It is negative if $\Pr(s_q^{(p)} \mid s_j^{(k)}) < \Pr(s_q^{(p)})$ and is zero if $\Pr(s_q^{(p)} \mid s_j^{(k)}) = \Pr(s_q^{(p)})$. $I(s_q^{(p)} : s_j^{(k)})$ intuitively measures the decrease (if positive) or increase (if negative) in uncertainty about the assignment of $s_q^{(p)}$ to g_q given that the state of g_j is $s_j^{(k)}$. Based on the mutual information, the weight of evidence provided by $s_j^{(k)}$ supporting or refuting the state of g_q is $s_q^{(p)}$ can be defined as follows:

$$W(\text{State} = s_q^{(p)} / \text{State} \neq s_q^{(p)} \mid s_j^{(k)}) \qquad (3.5)$$

$$= I(s_q^{(p)} : s_j^{(k)}) - I(\neq s_q^{(p)} : s_j^{(k)}).$$

Similar as before, $I(\neq s_q^{(p)} : s_j^{(k)})$ measures the decrease (if positive) or increase (if negative) in uncertainty about the assignment of the state, which is not equal to $s_q^{(p)}$, to g_q given that the state of g_j is $s_j^{(k)}$. In other words, *W* can be interpreted as a measure of the difference in the gain in information, $I(s_q^{(p)} : s_j^{(k)})$ - $I(\neq s_q^{(p)} : s_j^{(k)})$. The weight of evidence is positive if $s_j^{(k)}$ provides positive evidence supporting the assignment of $s_q^{(p)}$ to g_q; otherwise, it is negative. Since this measure is probabilistic, it can work effectively even when the data being dealt with contains incomplete, missing, or erroneous values.

Given a collection of the selected states of M' genes, where $M' < M$, that can be utilized to construct characteristic descriptions of $s_q^{(p)}$, the total weight of evidence supports the assignment of $s_q^{(p)}$ to g_q is defined as follows,

$$TW(\text{State} = s_q^{(p)} / \text{State} \neq s_q^{(p)} \mid s_1^{(k)} \ldots s_j^{(k)} \ldots s_{M'}^{(k)}) \qquad (3.6)$$

$$= \sum_{j=1, j\neq q}^{M'} \text{W}(\text{State} = s_q^{(p)} / \textit{State} \neq s_q^{(p)} \mid s_j^{(k)}).$$

Therefore, given a testing sample, the state, $s_q^{(p)}$, is inferred to the target gene, g_q, if the total weight of evidence is maximized.

With the association-discovery technique, interesting association relationships between genes can be discovered. In addition, this technique can also be used to discover interesting sequential associations from time-dependent gene expression data. In the following section, this sequential association discovery approach will be given.

3.2 Discovery of Sequential Associations from Time-dependent Gene Expression Data

Let us assume that we are given a set of time-dependent gene expression data, G, consisting of M time series collected from experiments with M genes. Each of these M time series consists, in turn, of N data points collected from N different experimental conditions, $E_1,...,E_{t-1},E_t,E_{t+1},...,E_N$, carried out, one after the other, at N different time instances. The data set, G, can therefore be represented as: $\mathrm{G}=\{g_1,\ldots,g_j,\ldots,g_M\}$, where each gene, say g_j, takes on the expression value, e_{tj}, under the experimental condition E_t. By computing an average gene expression value, $\bar{e}_j$, which serves as a reference point for how large the value is (as discussed in Section 3.1), instead of e_{tj}, g_j can now be considered as taking on a transformed expression value, $s_{tj}\in\{H,L\}$ under the experimental condition E_t. This process of transformation is also performed for each of the M genes in G (Fig. 3.1). Given the above representation, the association-discovery technique can be used to discover interesting sequential associations as follows [113]-[115].

To discover the sequential associations, the first step is to determine, for the target gene, g_q, which other genes, g_j, it is dependent on, where $q\neq j$ and $g_q\in G$. For gene regulatory relationships, if the expression of g_q is dependent on the expression of g_j, one would expect to observe that the expression levels of g_q be associated with those of g_j after a certain amount of time delay [15]. For this reason, we detect if there exists interesting sequential associations between the states of g_q in E_t with those of g_j in E_{t-1}.

If, say, "g_q is highly expressed in E_t" is dependent on "g_j is highly expressed in E_{t-1}" (in this case, $s_{tq} = s_q^{(p)} = H$, and $s_{(t-1)j} = s_j^{(k)} = H$, $s_q^{(p)}$ and $s_j^{(k)}$ are described in Section 3.1), then we can expect the observed total number of occurrences of "g_j is highly expressed in E_{t-1} and g_q is highly expressed in E_t" is significantly different from its expected total number of occurrences. To determine if this is the case, the adjusted residual discussed in Section 3.1 can be used. Table 3.1 shows the summary of the notations used.

TABLE 3.1
SUMMARY OF THE NOTATIONS USED

obs_{pk}	Observed total number of occurrences of: g_j is highly expressed in E_{t-1} and g_q is highly expressed in E_t
$\exp_{pk}$	Expected total number of occurrences under the assumption that: g_q is highly expressed in E_t is independent of whether g_j is highly expressed in E_{t-1}
obs_{p+}	Observed total number of occurrences of: g_q is highly expressed in E_t and g_j is highly expressed or lowly expressed in E_{t-1}
obs_{+k}	Observed total number of occurrences of: g_q is highly expressed or lowly expressed in E_t and g_j is highly expressed in E_{t-1}

Given obs_{pk} and $\exp_{pk}$, we can determine whether obs_{pk} is significantly different from $\exp_{pk}$ using the adjusted residual shown in Eq.(3.2). According to Section 3.1, this statistic, d_{pk}, has an approximate standard normal distribution and the sequential association "g_j is highly expressed in E_{t-1} and g_q is highly expressed in E_t" is interesting when the test statistic is statistically significant. In other words, if $d_{pk} > 1.96$, we can conclude, with a confidence level of 95 percent,

that "g_q is highly expressed in E_t" is dependent on "g_j is highly expressed in E_{t-1}" and this sequential association can be represented as $s_{(t-1)j}=H \rightarrow s_{tq}=H$.

It should be noted that there are two major types of gene regulatory relationships at the level of transcription [27], [154]. They are activation and inhibition. Activation and inhibition can take place through the regulator (the protein product of g_j) directly binding to g_q (the target gene), or by binding other regulators and thus controlling g_q indirectly. In the activation process, if one is hypothesizing that g_j activates g_q, one would expect to see in the data that, if the state of g_j is high, it is to be followed by the state of g_q being also high and if the state of g_j is low, it is to be followed by the state of g_q being low. The expectation would be reversed for inhibition. Hence, based on the sequential associations discovered between the various states of different genes, we may determine whether or not one gene is activated or inhibited by another.

Since the discovered sequential association is not completely deterministic, the uncertainty associated with $s_{(t-1)j}=H \rightarrow s_{tq}=H$ can be modeled with the confidence measure defined as $\Pr(s_{tq}=H \mid s_{(t-1)j}=H)$. For the purpose of predicting s_{tq} in a future time point, a weight of evidence measure, $W(s_{(t-1)j}=H \rightarrow s_{tq}=H)$, which is defined in terms of the mutual information $I(s_{tq}=H : s_{(t-1)j}=H)$, can be used as follows:

$$W(s_{(t-1)j}=H \rightarrow s_{tq}=H) \tag{3.7}$$

$$= W(s_{tq}=H / s_{tq} \neq H \mid s_{(t-1)j}=H)$$

$$= I(s_{tq}=H : s_{(t-1)j}=H) - I(s_{tq} \neq H : s_{(t-1)j}=H),$$

where

$$I(s_{tq}=H : s_{(t-1)j}=H) = \log \frac{\Pr(s_{tq}=H \mid s_{(t-1)j}=H)}{\Pr(s_{tq}=H)}.$$

$W(s_{(t-1)j}=H \rightarrow s_{tq}=H)$ measures the amount of positive or negative evidence that is provided by $s_{(t-1)j}=H$ supporting or refuting the state of the target gene, g_q, in E_t to be *H*. The sequential association discovery approach can be summarized as shown in Fig. 3.3. With the discovered sequential associations, whether or not the target gene will be highly or lowly expressed in a next time point can be predicted. In addition, they can also allow gene expression of the unseen samples to be predicted. This can be done as follows.

***Step 1*:** Compute the average expression value, $\bar{e}_j$, of each gene, g_j, in G, $j = 1,...,M$.

Step 2: Map each real-valued expression value, e_{tj}, into one of two states: H and L, depending on how close they are to $\bar{e}_j$.

Step 3: For the target gene, g_q, determine if there are interesting sequential associations between its states and those of other genes, g_j, using the adjusted residual.

Step 4: If a sequential association is statistically significant, then it is used to decide if g_q is activated or inhibited by another.

Step 5: Based on the discovered sequential associations and a weight-of-evidence measure, the state of g_q in the next time point can be predicted.

Fig. 3.3 Summary of the sequential association discovery approach.

Given a set of time-dependent expression data collected from a set of M' genes from the unseen samples (i.e., gene expression data that is not in the original database). This set of M' genes can be represented by $G' = \{g_{(1)},...,g_{(j)},...,g_{(M')}\}$

where $G' \subseteq G$ and $M' \leq M$. The gene expression values can then be mapped into states of *H* and *L* based on Steps 1 and 2 above (Fig. 3.3). To predict the expression state of the target gene, $g_{(q)}$, at time *t*, the discovered sequential associations can be searched to see which other genes $g_{(q)}$ is dependent on. If, say, the sequential association, $s_{(t-1)(j)} = H \rightarrow s_{t(q)} = H$ was previously discovered in the original database, then we can conclude that there is some evidence supporting $g_{(q)}$ to be highly expressed at time *t* if $g_{(j)}$ was highly expressed at the last time instance. By considering if and how $g_{(q)}$ is dependent on other genes in G', then we can combine the evidence that support $g_{(q)}$ to be highly expressed by computing a total weight of evidence measure. Suppose that $g_{(q)}$ is only dependent on some other genes, $g_{(1)},...,g_{(j)},...,g_{(\beta)}$, in G', where $\beta < M'$, then the total weight of evidence measure can be computed as follows:

$$TW(s_{(t-1)(1)} = S,...,s_{(t-1)(j)} = S,...,s_{(t-1)(\beta)} = S \rightarrow s_{t(q)} = H), \qquad (3.8)$$

where $S \in \{H, L\}$.

$$= TW(s_{t(q)} = H / s_{t(q)} \neq H \mid s_{(t-1)(1)} = S,...,s_{(t-1)(j)} = S,...,s_{(t-1)(\beta)} = S)$$

$$= \sum_{j=1}^{\beta} W(s_{t(q)} = H / s_{t(q)} \neq H \mid s_{(t-1)(j)} = S).$$

The total weight of evidence for $g_{(q)}$ to be highly or lowly expressed are computed respectively and the state for $g_{(q)}$ is determined by the one with the greatest total weight of evidence, i.e.,

$$\underset{S_{MAX} \in \{H,L\}}{MAX} TW(s_{(t-1)(1)} = S,...,s_{(t-1)(j)} = S,...,s_{(t-1)(\beta)} = S \rightarrow s_{t(q)} = S_{MAX}). \quad (3.9)$$

3.3 Experiments

A. Experimental Data

For our experiments, we used a set of real, time-dependent gene expression data. In this data set, the biological samples were synchronized by different methods such as α factor arrest, arrest of a cdc15, and cdc28 temperature-sensitive mutant. Using periodicity and correlation algorithms, a set of cell cycle regulated genes (from about 6000 genes) that meet an objective minimum criterion for cell cycle regulation were identified [141].

B. Evaluating Criteria

In our analysis, we chose the cdc15 experiment as the training set as it has the largest number of samples/experimental conditions (24 samples). The other two data sets, alpha experiment (18 samples) and cdc28 experiment (17 samples), were used as the testing sets. During the training process, interesting association relationships between the cell cycle regulated genes were discovered. For performance evaluation, we selected a subset of target genes from two important functional groups, cyclin and histone, for testing. This subset includes CLN1, HTA1, HTB1, CLB1, CLN2, HTA2, HTB2, CLB2 and CLB6 (Table 3.2). Based on the discovered sequential associations, the states of these genes in independent testing sets were predicted and then compared with what are known about them.

TABLE 3.2
SUMMARY OF TARGET GENES SELECTED. PEAK EXPRESSION OF THE TARGET GENES COVER THE STANDARD PHASES OF CELL CYCLE (G1, S, G2 AND M)

Standard Name	*Systematic Name*	*Peak*
CLN1	YMR199W	G1
HTA1	YDR225W	S
HTB1	YDR224C	S
CLB1	YGR108W	G2/M
CLN2	YPL256C	G1
HTA2	YBL003C	S
HTB2	YBL002W	S
CLB2	YPR119W	G2/M
CLB6	YGR109C	G1

3.4 Results and Discussions

During the training process, the sequential associations that can be used to construct characteristic descriptions of the states of each target gene were discovered. The examples of such sequential associations discovered are given in Figs. 3.4 and 3.5. For example, the first sequential association shown in Fig. 3.4 reveals that if the state of RME1 is high (H), then the state of CLN2 is high (H) in the next time instance. This also implies that CLN2 is activated by RME1.

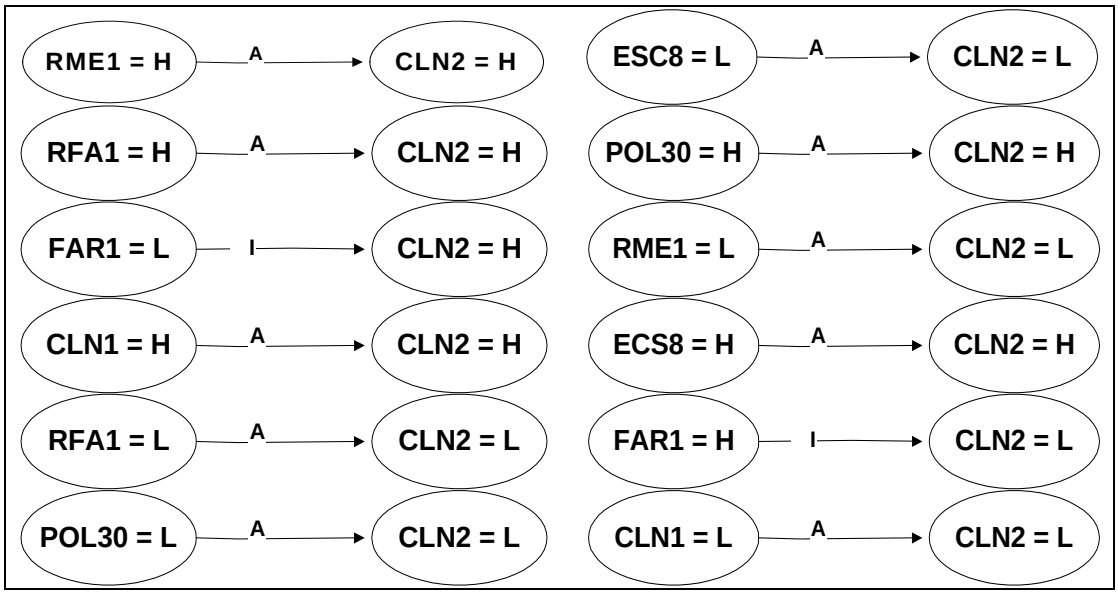

Fig. 3.4 Interesting sequential associations that can be used to construct characteristic descriptions of the states of CLN2 gene (A is activation and I is inhibition).

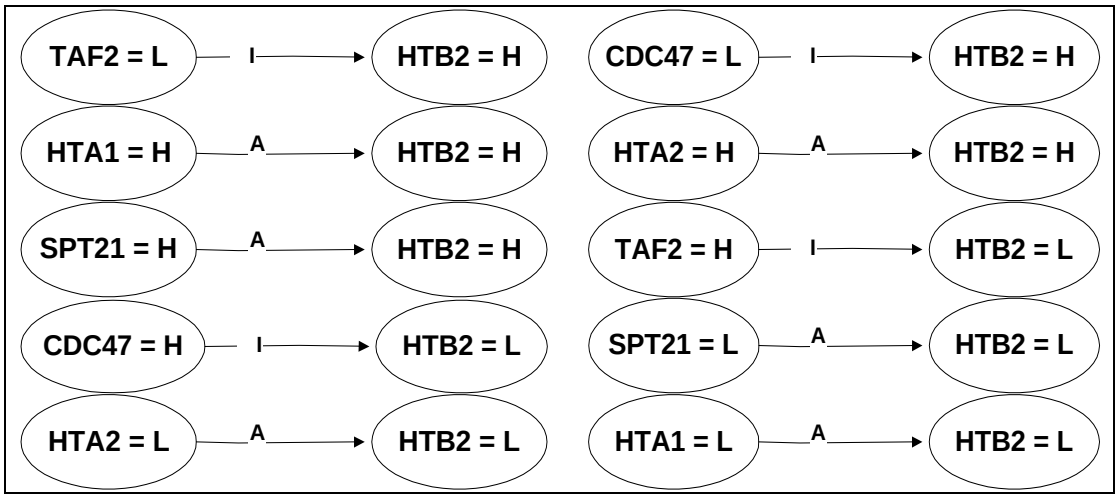

Fig. 3.5 Interesting sequential associations that can be used to construct characteristic descriptions of the states of HTB2 gene.

Based on the sequential associations discovered, the states of each target gene in the testing sets (alpha and cdc28) can be predicted. The predicted states were then compared with the original states of the target gene and the average prediction accuracy can be calculated in each testing set.

For experimentation, we evaluated the performance of the proposed association-discovery technique by comparing it to the well-known decision-tree based classification algorithm, C4.5 [131], [159]. We chose C4.5 because, compared to other approaches which can only infer plausible relationships between genes, it, like the proposed technique, also has both predictive and explanatory capability (as discussed in Chapter 2, Section 2.3.2).

To predict the states of the target genes with C4.5, the expression values of each gene were divided into two different states as discussed before: "highly expressed (H)", and "lowly expressed (L)". The expression value, e_{tj}, of each gene can then be mapped to H if $e_{tj} > \bar{e}_j$, and L if $e_{tj} \leq \bar{e}_j$, where $\bar{e}_j$ is the average expression value of g_j under all the experimental conditions.

In order to improve the performance of C4.5, feature selection has been performed for it. Many feature selection methods have been proposed to reduce the number of attributes in gene expression data. The most popular methods are the filter and wrapper methods [49], [65], [143], [146], [160]. Based on the *t*-statistic [143], [146], for each target gene, the following steps were adopted and performed to select genes (attributes) for C4.5: (i) all cell cycle regulated genes are sorted in descending order of their *t*-values, (ii) initially, the top 5% of the genes are selected and removed from the ranked list, (iii) using C4.5 with a 10-fold cross validation on the training data of this subset of selected genes, the average classification accuracy is obtained, (iv) an additional 5% of genes from the ranked list are then added into the subset of selected genes, (v) steps (iii) and (iv) are repeated until the classification performance converged, then (vi) the final subset of genes with the highest classification accuracy is selected.

In Tables 3.3 and 3.4, the comparisons of the average prediction accuracy of different approaches are showed. The results shown in the tables indicate that the proposed technique has higher prediction accuracy than C4.5 and the feature selection version of it for both testing sets. Even though C4.5 and the feature selection version of it already performed quite well, the results seem to indicate that the association-discovery technique is even more effective.

TABLE 3.3
COMPARISON OF THE AVERAGE PREDICTION ACCURACY (ALPHA DATASET) (FS REPRESENTS FEATURE SELECTION)

Standard Name	*Proposed*	*C4.5*	*C4.5 + FS*
CLN1	88.89%	66.67%	83.33%
HTA1	88.89%	61.11%	77.78%
HTB1	94.44%	66.67%	77.78%
CLB1	88.89%	66.67%	83.33%
CLN2	88.89%	66.67%	77.78%
HTA2	88.89%	72.22%	83.33%
HTB2	83.33%	61.11%	72.22%
CLB2	83.33%	61.11%	77.78%
CLB6	88.89%	72.22%	83.33%
Average	88.27%	66.05%	79.63%

TABLE 3.4
COMPARISON OF THE AVERAGE PREDICTION ACCURACY (CDC28 DATASET)

Standard Name	*Proposed*	*C4.5*	*C4.5 + FS*
CLN1	82.35%	64.71%	76.47%
HTA1	82.35%	58.82%	70.59%
HTB1	82.35%	52.94%	64.71%
CLB1	88.24%	70.59%	82.35%
CLN2	88.24%	70.59%	82.35%
HTA2	82.35%	64.71%	76.47%
HTB2	82.35%	58.82%	70.59%
CLB2	88.24%	70.59%	88.24%
CLB6	88.24%	64.71%	82.35%
Average	84.97%	64.05%	77.12%

Biological Interpretation:

In order to evaluate the biological significance of the discovered sequential associations, we tried to verify that any known gene regulatory relationships [19], [61], [104] could be revealed from them. Fig. 3.6 shows some biologically meaningful sequential associations discovered by the proposed technique. Based on these findings, we can then construct the gene interaction diagram [15] as shown in Fig. 3.7. This diagram might provide important clues in inferring the structures of GRNs. For comparison, we also show the interaction diagram (Fig. 3.8) that was

constructed based on the rules discovered by C4.5 (the best diagram, which reveals more known gene regulatory relationships, among C4.5 and C4.5 with feature selection). Comparing the diagrams, it should be noted that some of the known gene regulatory relationships can only be discovered by the proposed technique.

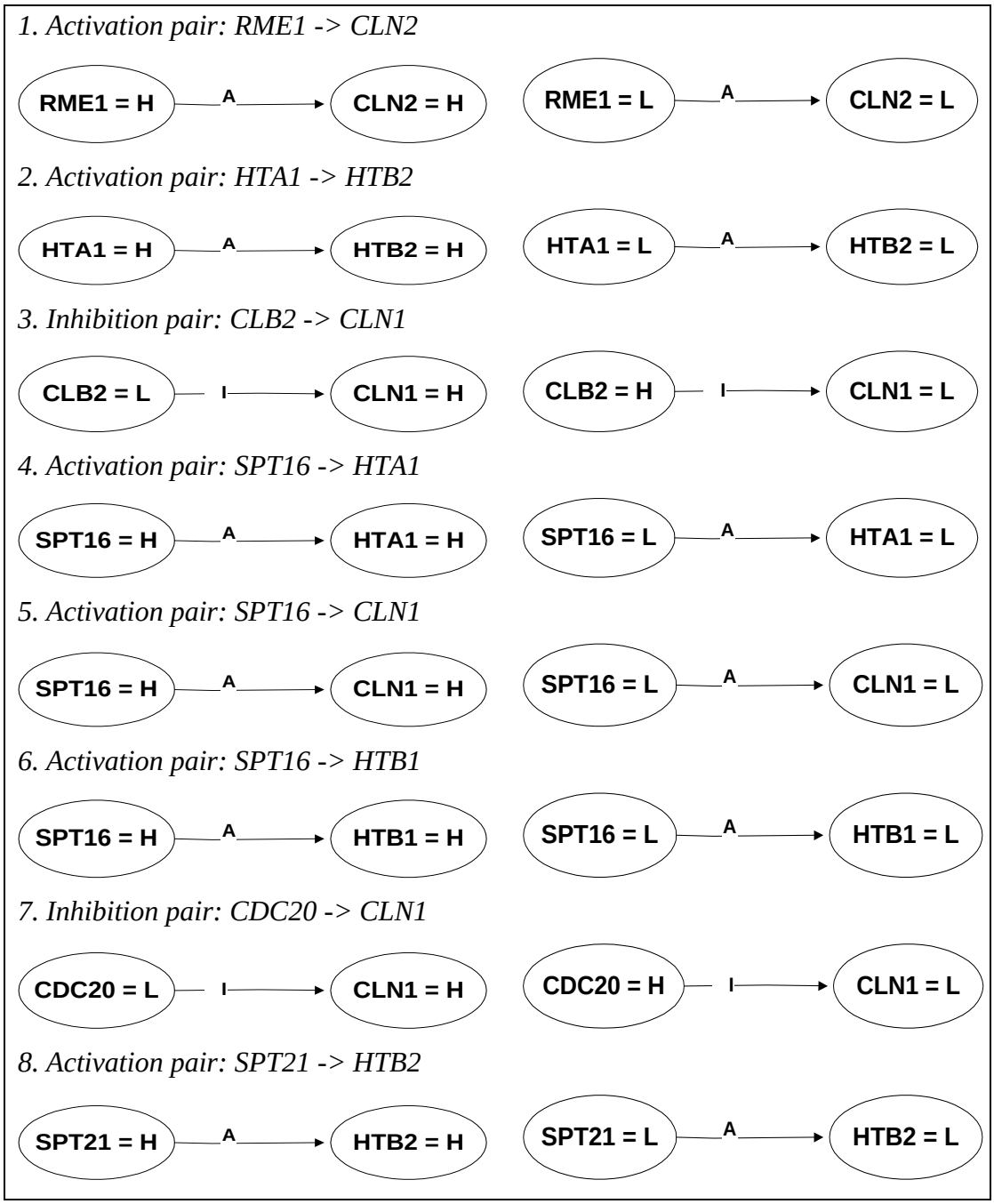

Fig. 3.6 Biologically meaningful sequential associations discovered by the proposed technique (to be continued).

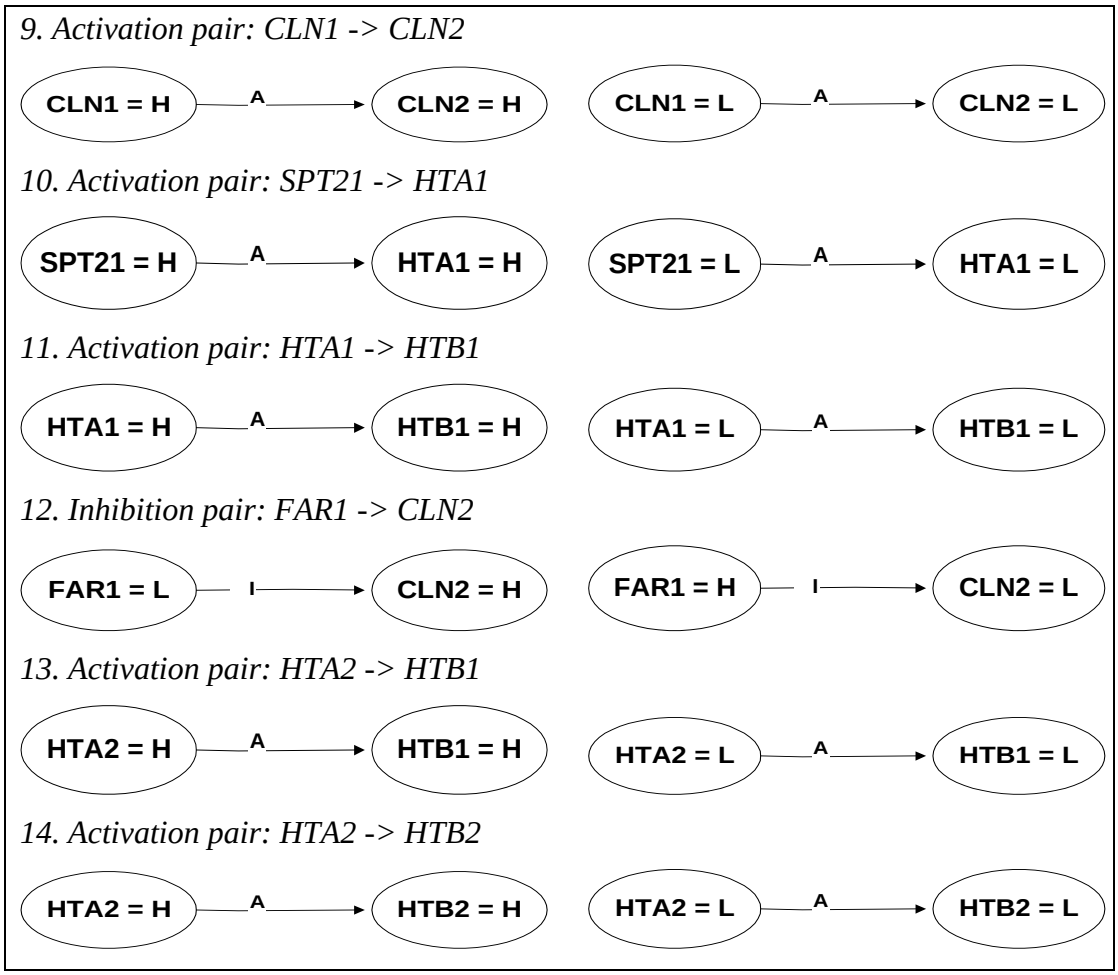

Fig. 3.6 Biologically meaningful sequential associations discovered by the proposed technique.

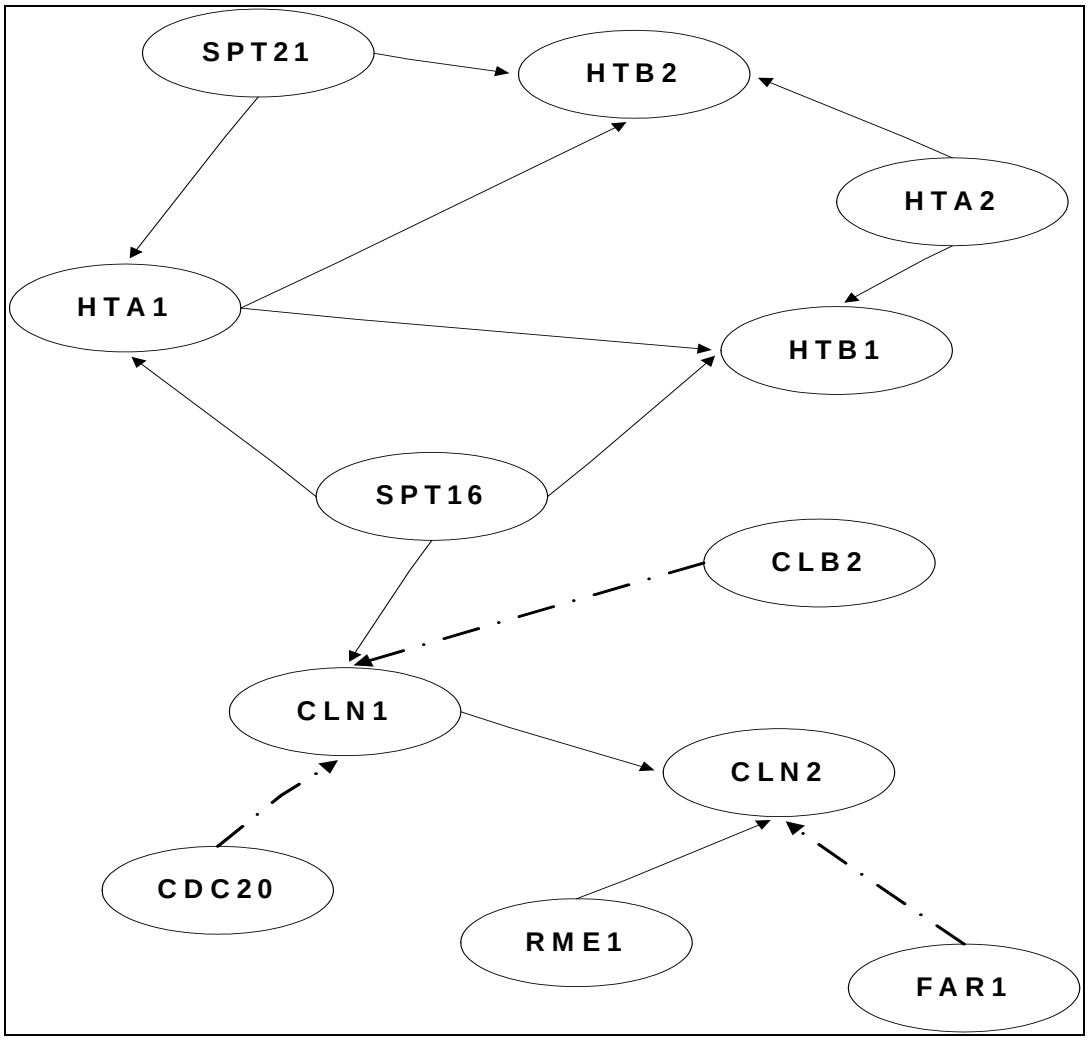

Fig. 3.7 Gene interaction diagram constructed by the known gene regulatory relationships discovered by the proposed technique (12 genes and 14 known regulatory relationships involved). Solid lines correspond to activation relationships and broken lines correspond to inhibition relationships.

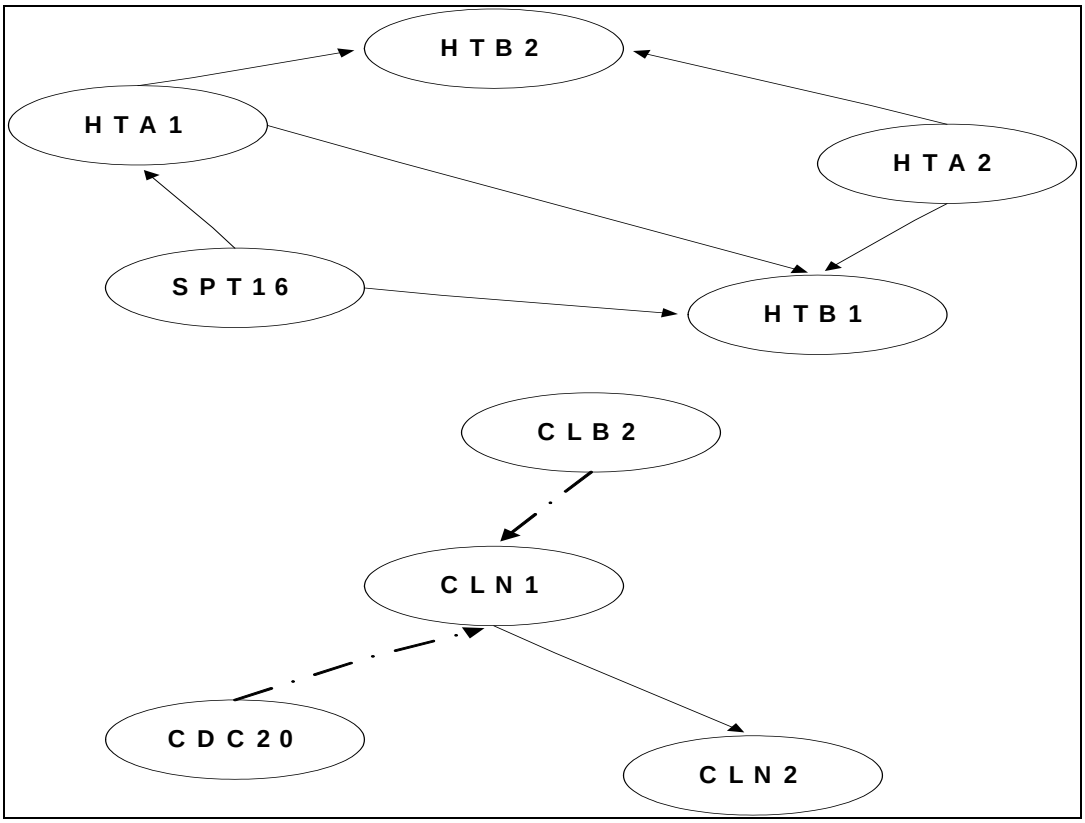

Fig. 3.8 Gene interaction diagram constructed by the known gene regulatory relationships discovered by C4.5 (9 genes and 9 known regulatory relationships involved).

In this chapter, we have proposed to use an association-discovery technique for the reconstruction of GRNs from time-dependent gene expression data. This technique can discover interesting association relationships between genes in high-dimensional and very noisy data without the need for additional feature selection procedures. Based on the discovered sequential associations, the user can not only determine those genes affecting a target gene but also can identify whether or not the target gene is supposed to be activated or inhibited. In addition, the sequential associations discovered can also be used to predict how a gene would be affected by other genes from the unseen samples. Experimental results on real expression data show that the proposed technique can be very effective and the discovered sequential associations reveal known gene regulatory relationships that could be used to infer the structures of GRNs. One additional advantage of the reconstruction

of GRNs using the association-discovery technique is that the user can easily improve the classifier by adding new expression data and reproduce underlying structures of a network consistent with the data. Since such iterative improvements can be part of an interactive process, therefore, the proposed technique can be considered as a basis for an interactive expert system for GRNs reconstruction.

Chapter 4

Clustering and Re-clustering of Gene Expression Data

In clustering gene expression data [17], [52], [74], [148]-[149], various forms of inaccuracies and data variations need to be reduced as noise can be introduced at different stages - the production of the arrays, preparation of the samples, hybridization experiments and extraction of the hybridization results (as discussed in Chapter 2, Section 2.3.3). Genetic variations and impurity of tissue samples may also introduce additional difficulties in the analysis [16]. To cluster gene expression data in the presence of these various types of noise and overcome some limitations of existing clustering algorithms, in this chapter, we propose a two-phase clustering algorithm [112], [118], [120] consisting of an initial clustering phase and a second re-clustering phase.

In the first phase, local information is extracted for the clustering process by computing a pair-wise distance measure between gene expression profiles. This information is then used for a conservative clustering approach that prefers to leave uncertain genes unassigned rather than forcing them into one of the clusters thereby rendering the discovered clusters less reliable. The second phase consists of a re-clustering process. For the purpose of re-clustering, global information is obtained through the discovery of interesting association relationships between gene expression levels and cluster labels. In doing so, the association-discovery technique discussed in Chapter 3 (Section 3.1) is used to discover interesting associations by

differentiating among the expression levels that are relevant for the clustering process from those that are irrelevant. If an expression level is relevant in determining whether or not a gene should belong to a particular cluster, then it is reflected by the interestingness measure. Since the interestingness measure is probabilistic, it can work effectively even when the data being dealt with contains incomplete, missing, or even erroneous values. Once the associations are discovered, they can be made explicit for possible interpretation. These associations specify different characteristics, in terms of what expression levels the genes should have under a particular set of experimental conditions, that different clusters of genes possess and they can be easily understood and interpreted by human users. In the following section, the proposed clustering algorithm will be presented in details.

4.1 A Two-Phase Clustering Algorithm

To describe the proposed two-phase clustering algorithm [112], [118], [120], let us assume that we are given a set of gene expression data, *G*, consisting of the data collected from *N* genes in *M* experiments carried out under different experimental conditions. Let us represent the data set as a set of *N* genes, $\mathbf{G} = \{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_i, \ldots, \boldsymbol{g}_N\}$, with each gene, $\boldsymbol{g}_i$, $i = 1, ..., N$, characterized by *M* different experimental conditions, $E_1, \ldots, E_j, \ldots, E_M$, whose values, $e_{i1}, \ldots, e_{ij}, \ldots, e_{iM}$, where e_{ij} represents the expression value of the i^{th} gene under the j^{th} experimental condition (Fig. 4.1).

	E_1	...	E_j	...	E_M
g_1	e_{11}	...	e_{1j}	...	e_{1M}
⋮	...	...	...	...	...
g_i	e_{i1}	...	e_{ij}	...	e_{iM}
⋮	...	...	...	...	...
g_N	e_{N1}	...	e_{Nj}	...	e_{NM}

Fig. 4.1 Gene expression data representation.

Phase 1 - The cluster initialization phase:

To find the initial clusters, the popular hierarchical agglomerative clustering algorithm [156] is used. This algorithm consists of a series of successive fusions of N genes, $\boldsymbol{g}_1,\ldots,\boldsymbol{g}_i,\ldots,\boldsymbol{g}_N$, into clusters and the results of this fusion process are presented in the form of what is called a dendrogram. A dendrogram displays the results after each successive fusion. The genes falling along a particular branch in a dendrogram form a cluster and the similarity between different clusters are also shown. By cutting the dendrogram at some level, a specific number of clusters can be obtained. The advantage of using hierarchical clustering algorithm is that a suitable cutoff level based on prior knowledge can be used. As a result, a set of more reliable initial clusters can be obtained (as discussed before, in this phase, it prefers to leave uncertain genes unassigned rather than forcing them into one of the clusters thereby rendering the discovered clusters less reliable.). Since the Pearson correlation coefficient is more commonly used for gene expression data and is known to be better than the Euclidean distance in dealing with noise [16]. It is therefore used as a similarity function in this phase. For any two genes, g_x and g_y, whose expressions are monitored over a series of M different experimental conditions, the measure is defined as follows:

$$S(g_x, g_y) = \frac{1}{M}\{\sum_{j=1}^{M}(\frac{e_{xj} - \bar{E}_x}{\Theta_x})(\frac{e_{yj} - \bar{E}_y}{\Theta_y})\}, \quad (4.1)$$

where

$$\Theta_x = \sqrt{\sum_{j=1}^{M}\frac{(e_{xj} - \bar{E}_x)^2}{M}}, \Theta_y = \sqrt{\sum_{j=1}^{M}\frac{(e_{yj} - \bar{E}_y)^2}{M}}, \bar{E}_x = \frac{\sum_{j=1}^{M} e_{xj}}{M}, \bar{E}_y = \frac{\sum_{j=1}^{M} e_{yj}}{M}.$$

Phase 2 - The re-clustering phase:

With the proposed association-discovery technique (as discussed in Chapter 3, Section 3.1), in this phase, genes that were not assigned to any clusters in Phase 1 are assigned and those that have already been assigned are re-evaluated to determine if they should remain in the same cluster or be assigned to a different one. This assignment process is performed in two steps as follows. Firstly, interesting associations are identified in the initial clusters that are statistically significant. Then, based on the discovered associations, all the genes that were previously unassigned to any clusters are assigned to one of the clusters, and the cluster memberships of those that were previously assigned are also re-evaluated. The details of this assignment process are given below.

Step 1 - Discovering interesting associations between gene expression levels and cluster labels:

To minimize the effect of noise in the re-clustering process, rather than the actual expression values, the data is partitioned into intervals (levels) instead. The partitioning, which is also called discretization, is based on a popular technique as described in [35] so as to minimize the loss of information during the process.

After discretization, interesting association relationships are discovered in each initial cluster by detecting for the associations between the expression levels of the genes that belong to a particular cluster and the cluster label itself. To do so, we let obs_{pk} be the observed total number of genes, $g_1,...,g_i,...,g_l$, where $l \le N$, in the data that belong to a given cluster, C_p, where $p = 1,...,P$ and P is the total number of initial clusters discovered, and are characterized by the expression values that are within the interval of $e_j^{(k)}$ (the expression values, $e_{1j},...,e_{ij},...,e_{lj}$, are within the interval of $e_j^{(k)}$. In other words, the genes, $g_1,...,g_i,...,g_l$, have the expression level $e_j^{(k)}$), where $k = 1,...,K_{val}$, and K_{val} is the total number of distinct data intervals (expression levels) of E_j. We also let $\exp_{pk} = \frac{obs_{p+} obs_{+k}}{N'}$ be the expected total under the assumption that being a member of C_p is independent of whether or not a gene has the characteristic $e_j^{(k)}$, where $obs_{p+} = \sum_{k=1}^{K_{val}} obs_{pk}$, $obs_{+k} = \sum_{p=1}^{P} obs_{pk}$ and $N' = \sum_{p,k} obs_{pk} \le N$ due to possible missing values in the data. An association is then considered interesting if obs_{pk} is significantly different from $\exp_{pk}$. To determination if this is the case, the adjusted residual as shown in Eq. (3.2) is used. According to Section 3.1, this statistic has an approximate standard normal distribution and an association is considered to be interesting if the test statistic is statistically significant.

Step 2 - Assignment and re-assignment of genes:

Using the adjusted residual, we can determine if $e_j^{(k)}$, under the experimental condition, E_j, is associated with a cluster, C_p, say, a 95% confidence level ($d_{pk} > 1.96$). If it is the case, then it can be utilized to construct characteristic description of C_p. This description is represented as follows: If the expression value of a gene in E_j is within the interval of $e_j^{(k)}$, then it is with certainty *W*(Cluster = C_p / Cluster $\neq C_p$ | $e_j^{(k)}$) that the gene belongs to C_p, where *W*, the weight of evidence measure [124], is defined in terms of the mutual information $I(C_p : e_j^{(k)})$ as follows:

$$W(\text{Cluster} = C_p \text{ / Cluster} \neq C_p \mid e_j^{(k)}) \qquad (4.2)$$

$$= I(C_p : e_j^{(k)}) - I(\neq C_p : e_j^{(k)}),$$

where

$$I(C_p : e_j^{(k)}) = \log \frac{\Pr(C_p \mid e_j^{(k)})}{\Pr(C_p)}.$$

The weight of evidence measures the amount of positive or negative evidence that is provided by $e_j^{(k)}$ supporting or refuting the labeling of a gene as C_p.

To re-evaluate the cluster membership of a gene, g_i, characterized by $E_1,\ldots,E_j,\ldots,E_M$, its description can be matched against the discovered associations. If the expression value, e_{ij}, of g_i satisfies the associations (i.e., the expression value, e_{ij}, of g_i is within the interval of $e_j^{(k)}$) that implies C_p, then we can conclude that the description of g_i partially matches that of C_p. By repeating the above

procedure, that is, by matching each expression value, e_{ij}, $j = 1,...,M$, of g_i against the discovered associations, the total weight of evidence of assigning g_i to C_p can be determined. Suppose that of the M characteristics that describe g_i, only M', $M' \leq M$, of them are found to match with the discovered associations. Then, the total weight of evidence supports the labeling of g_i as C_p is defined as follows,

$$TW(\text{Cluster} = C_p \ / \ \text{Cluster} \neq C_p \mid e_1^{(k)} ... e_j^{(k)} ... e_{M'}^{(k)}) \qquad (4.3)$$

$$= \sum_{j=1}^{M'} W(Cluster = C_p / Cluster \neq C_p \mid e_j^{(k)}).$$

Then, the cluster label C_p is inferred to a gene if:

$$\underset{p=1}{\overset{P}{\text{MAX}}} \{ \sum_{j=1}^{M'} W(Cluster = C_p / Cluster \neq C_p \mid e_j^{(k)}) \}. \qquad (4.4)$$

The re-clustering phase described above allows for probabilistic associations to be detected. It performs its task by distinguishing between relevant and irrelevant expression levels and by doing so, it takes into consideration global information contained in a specific cluster arrangement by evaluating the importance of different expression levels in determining cluster memberships. This feature makes the proposed algorithm more robust to noisy data when compared to those algorithms that only rely on local pair-wise similarity measures.

4.2 Experiments

A. Experimental Data

For experimentation, we used a set of simulated data consisting of 300 records each characterized by 50 different attributes that take on values from [0.0, 1.0]. Initially,

all these records were sampled from a uniform distribution and they were pre-classified into one of three clusters so that each cluster contains 100 records. To embed hidden patterns in the data, 10% of the attributes in each cluster were randomly selected. For each selected attribute, 40% of its values in that cluster were randomly generated from within a certain range [*L*, *U*], where $0.0 \leq L \leq U \leq 1.0$, so that *L* was selected uniformly from [0.0, 1.0] first, and *U* was then also selected uniformly from [*L*, 1.0]. Since this kind of data generation is non-deterministic, it is very difficult to discover the hidden regularities in this noisy data solely relying on the local pair-wise distances between records. Therefore, it is used to evaluate the effectiveness of the proposed algorithm.

In addition to the simulated data, we also used a set of real expression data. According to [141], the authors successfully applied a hierarchical clustering algorithm [52], [156] to cluster about 230 genes into 8 distinct clusters based on the similarity of expression profiles (under 77 different experimental conditions) and their prior biological knowledge. However, due to variation of the expression data, the cluster memberships of the remaining genes (about 560 genes) cannot be determined. Since the unassigned genes may have similar features to those that have been clustered successfully, it is important that they could also be properly assigned so that the clustering results can be better interpreted.

B. Evaluating Criteria

For performance evaluation, the proposed algorithm was evaluated based on three objective measures: (i) the F-measure, (ii) the predictive power measure, and (iii) the Davies-Bouldin validity index (DBI) measure.

The F-measure [96], which is typically used for cluster evaluation, combines the

idea of "precision" and "recall" from the field of information retrieval [86]. When the correct clustering arrangement of a set of data is known (i.e., in case of the simulated data described above), the F-measure can be used to determine how well a discovered clustering arrangement compares with that of the correct, original one. According to the F-measure, the records (genes) in a discovered cluster, C_q, can be considered as if they have been retrieved through a certain query. These records can then be compared against those in one of the original clusters, C_p, which can be considered as consisting of records desired by a user by posting the query. Given a set of discovered and original clusters, the F-measure is therefore defined as: $F(C_p, C_q) = \frac{2\,\mathrm{Re}call(C_p, C_q)\,\mathrm{Pr}ecision(C_p, C_q)}{\mathrm{Re}call(C_p, C_q) + \mathrm{Pr}ecision(C_p, C_q)}$, where $\mathrm{Re}call(C_p, C_q) = \frac{count_{C_pC_q}}{count_{C_p}}$ and $\mathrm{Pr}ecision(C_p, C_q) = \frac{count_{C_pC_q}}{count_{C_q}}$ and $count_{C_pC_q}$ is the number of records with cluster label C_p in the discovered cluster C_q, $count_{C_p}$ is the number of records with cluster label C_p and $count_{C_q}$ is the number of records in the discovered cluster C_q. The F-measure has value in the interval [0,1] and the larger its value, the better the clustering quality it reflects.

The predictive power measure is actually a measure of classification accuracy. If the clusters discovered are valid and of good qualities, we should expect patterns to be discovered in them. If these patterns are used to classify some testing data (i.e., data that is not in the original database), the classification accuracy can reflect how valid and how good the qualities of the discovered clusters are. In order to determine the classification accuracy, a set of training data can be randomly selected from each

cluster to construct a decision-tree classifier using C4.5 [131]. C4.5 is a greedy algorithm that recursively partitions a set of training data by selecting attributes that yield a maximum information gain measure at each step in the tree-construction process. After a decision tree is built, the cluster memberships of those genes that were not selected for training are then predicted. The percentage of accurate prediction can then be determined as classification accuracy. This accuracy measure is also referred to as the predictive power measure. If a clustering algorithm is effective, the discovered clusters should contain hidden patterns that can be used to accurately predict the cluster membership of the testing data. And if this is the case, the predictive power of a cluster grouping should be high. Otherwise, if a clustering algorithm is ineffective, the clusters it discovers are not expected to contain too many hidden patterns and the grouping is more or less random. And if this is the case, the predictive power is expected to be low. Hence, the greater the predictive power, the more interesting a cluster grouping is and vice versa. In our experiments, the predictive power measure was computed based on a ten-fold cross validation approach. For each fold, 90% of the genes in each cluster were randomly selected for training and the remaining 10% used for testing. After ten experiments corresponding to the ten folds were performed, the average predictive power of the discovered clusters was computed as the average classification accuracy over the ten experiments [159].

The DBI measure [47] is a function of the inter- and intra-cluster distances. These distances are considered good indicators of the quality of a cluster grouping as a good grouping should be reflected by a relatively large inter-cluster distance and a relatively small intra-cluster distance. In fact, many optimization clustering algorithms are developed mainly to maximize inter-cluster and minimize intra-

cluster distances. The DBI measure combines these two distances in a function to measure the average similarity between a cluster and its most similar one. Assume that a cluster grouping consisting of *k* clusters has been formed. Its DBI measure is then defined as follows [123]: $DBI = \frac{1}{k}\sum_{p=1}^{k}\max_{p\neq q}\left\{\frac{d_{intra}(C_p)+d_{intra}(C_q)}{d_{inter}(C_p,C_q)}\right\}$, where $d_{intra}(C_q) = \frac{\sum_{i=1}^{n_q}\| g_i - g_{qc} \|}{n_q}$, $d_{inter}(C_p,C_q) = \| g_{pc} - g_{qc} \|$, k denotes the total number of clusters, d_{intra} and d_{inter} denote the centroid intra- and inter-cluster distances respectively and n_q is the number of genes in a cluster, C_q. The intra-cluster distance for a given cluster is therefore defined to be the average of all pair-wise distances between the genes in C_q and its centroid, g_{qc} and the inter-cluster distance between two clusters, C_p and C_q, is computed as the distance between their centroids, g_{pc} and g_{qc}. A low value of DBI indicates good cluster grouping.

4.3 Results and Discussions

As discussed before, the proposed algorithm consists of two phases: an initial clustering phase and a second re-clustering phase. To find the initial clusters, we used the popular hierarchical agglomerative clustering algorithm as discussed in Section 4.1. The effectiveness of the proposed algorithm was evaluated according to the objective measures as discussed in Section 4.2. For comparison, we also show the statistic for the three other clustering algorithms popularly used to cluster gene expression data. They are the hierarchical clustering algorithm [156], the *k*-means

algorithm [100] and SOM [85] (as discussed in Chapter 2, Section 2.3.3). Since the Pearson correlation coefficient is more commonly used for gene expression data and is known to be better than the Euclidean distance in dealing with noise [16]. It was therefore used as a similarity function for these algorithms. Also, to ensure that the best results for the *k*-means algorithm and SOM were obtained, 100 runs were performed for each of them with each run using different randomly-generated initial cluster centroids. Only the best result from among these 100 runs was recorded. Table 4.1 shows the comparison of the average F-measure using the simulated data. Based on the results, it appears that the performances of these existing algorithms can be further improved. They do not seem to be able to discover hidden regularities effectively in very noisy data.

TABLE 4.1
COMPARISON OF THE AVERAGE F-MEASURE
(SIMULATED DATA)

	Proposed Re-clustered Hierarchical	*Hierarchical*	*k-means*	*SOM*
Average	0.77	0.58	0.63	0.55

In order to demonstrate the effectiveness of the re-clustering phase in the proposed algorithm, we have also applied it to the *k*-means algorithm and SOM by using them separately, instead of the hierarchical agglomerative algorithm, in the first cluster initialization phase. By repeating the same experiments, the performances of the re-clustered *k*-means and SOM algorithms are given below in Table 4.2.

TABLE 4.2
THE AVERAGE F-MEASURE OF THE PROPOSED ALGORITHM USING
K-MEANS AND SOM IN THE CLUSTER INITIALIZATION PHASE
(SIMULATED DATA)

	Re-clustered k-means	*Re-clustered SOM*
Average	0.73	0.71

The above experimental results show that the proposed algorithm is rather robust in the presence of a very noisy environment. It is able to perform better than the three popular clustering algorithms. When applying the re-clustering phase to *k*-means and SOM, it can also improve their performances. The reason, as stated earlier, is due to the proposed algorithm not only able to consider local information but also global information as reflected by the cluster arrangement. By making use of the association-discovery technique, it is able to distinguish between interesting and uninteresting expression levels and to discover hidden regularities that are not completely deterministic. Using real expression data, we have also performed experiments to evaluate the performances of different clustering algorithms and the results are showed in Tables 4.3-4.6.

TABLE 4.3
COMPARISON OF THE AVERAGE PREDICTIVE POWER
(GENE EXPRESSION DATA)

	Proposed Re-clustered Hierarchical	*Hierarchical*	*k-means*	*SOM*
Average	76.98%	63.11%	62.58%	56.83%

TABLE 4.4
THE AVERAGE PREDICTIVE POWER OF THE PROPOSED ALGORITHM USING K-MEANS AND SOM IN THE CLUSTER INITIALIZATION PHASE
(GENE EXPRESSION DATA)

	Re-clustered k-means	*Re-clustered SOM*
Average	69.90%	65.06%

TABLE 4.5
COMPARISON OF THE AVERAGE DBI
(GENE EXPRESSION DATA)

	Proposed Re-clustered Hierarchical	*Hierarchical*	*k-means*	*SOM*
Average	1.69	1.73	1.72	1.76

TABLE 4.6
THE AVERAGE DBI OF THE PROPOSED ALGORITHM USING
K-MEANS AND SOM IN THE CLUSTER INITIALIZATION PHASE
(GENE EXPRESSION DATA)

	Re-clustered *k*-means	*Re-clustered SOM*
Average	1.70	1.73

The performance of the proposed algorithm is again consistently better than other clustering algorithms. This indicates that it is more robust in the presence of noisy data collected under real experiments. Moreover, by using the *k*-means algorithm and SOM in the first phase and then applying the re-clustering phase in the second, we can see some level of performance gain as well. With the proposed algorithm, we discovered the same 8 initial clusters as described in Section 4.2 (part A) after Phase 1. These clusters have characteristics as shown in Table 4.7 below. In the re-clustering phase (Phase 2), we successfully classified the remaining unclassified genes. The re-clustering results are also showed in Table 4.7.

TABLE 4.7
SUMMARY OF THE INITIAL AND RE-CLUSTERED CLUSTERS
(GENE EXPRESSION DATA)

Cluster ID	*Cluster name*	*Peak expression*	*No. of genes (Initial cluster)*	*No. of genes (Re-clustered cluster)*
C1	CLB2	M	32	168
C2	CLN2	G1	57	332
C3	Histone	S	9	13
C4	MAT	M/G1	13	25
C5	MCM	M/G1	38	116
C6	MET	S	20	41
C7	SIC1	M/G1	27	54
C8	Y’	G1	31	43
	Total no.		227	792

Figs. 4.2-4.5 show the microarray images of two initial and re-clustered clusters: CLB2 and MCM. The figures show the same cluster before and after the re-clustering phase. Using the hierarchical approach there are only initially 32 genes in

CLB2. After re-clustering, there are 168 genes. Similarly, for MCM, the number has increased from 38 to 116.

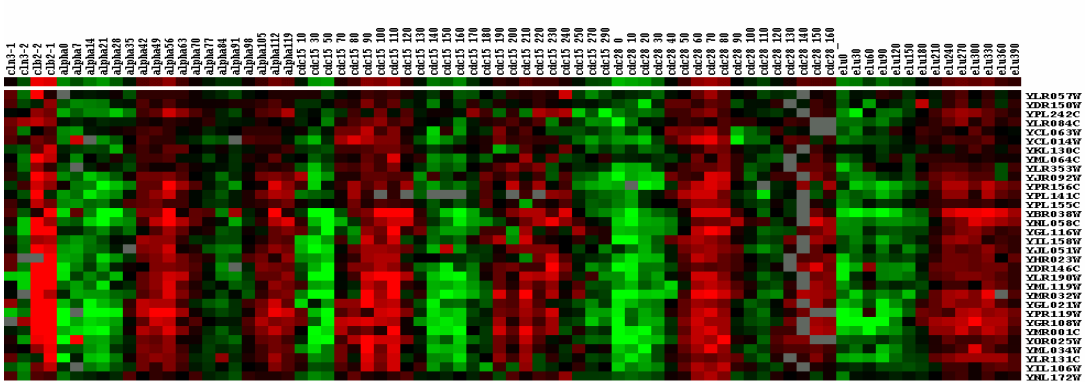

Fig. 4.2 CLB2 initial cluster (32 genes) (each label on the top indicates the experimental condition, and each label on the right indicates the gene name).

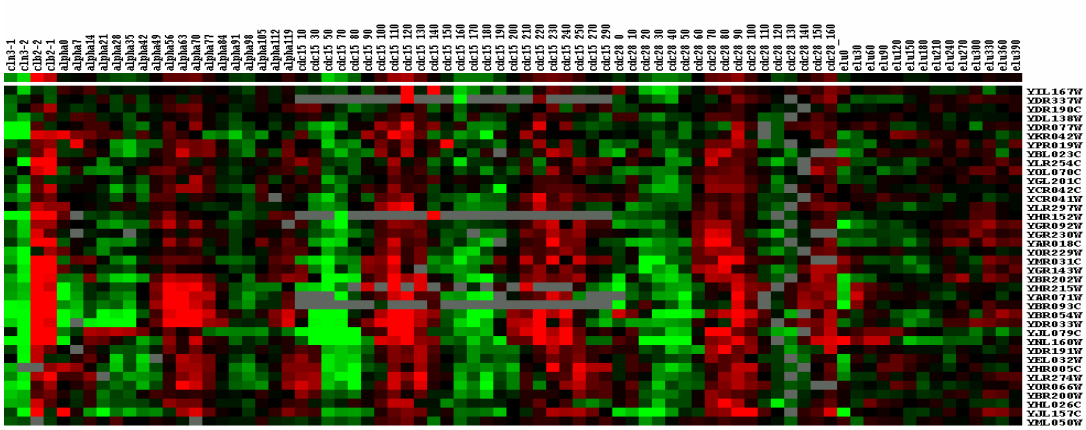

Fig. 4.3 MCM initial cluster (38 genes).

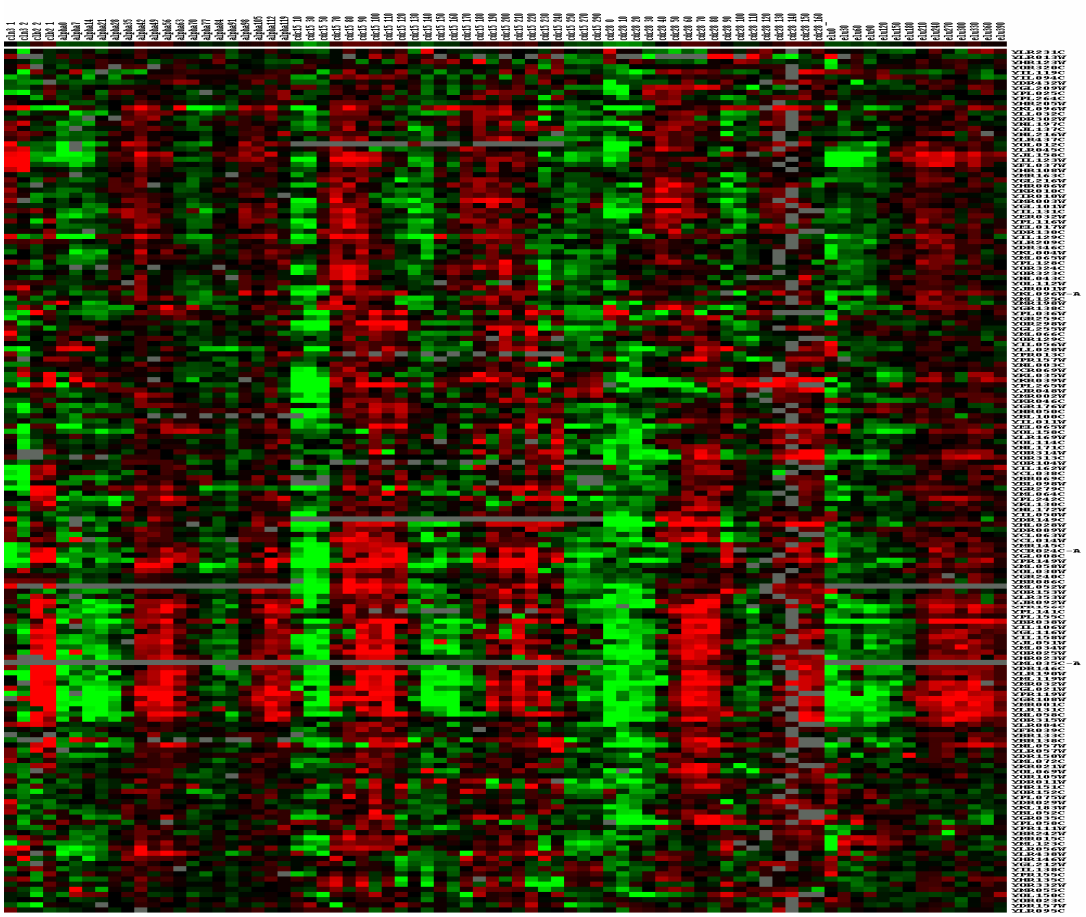

Fig. 4.4 CLB2 re-clustered cluster (168 genes).

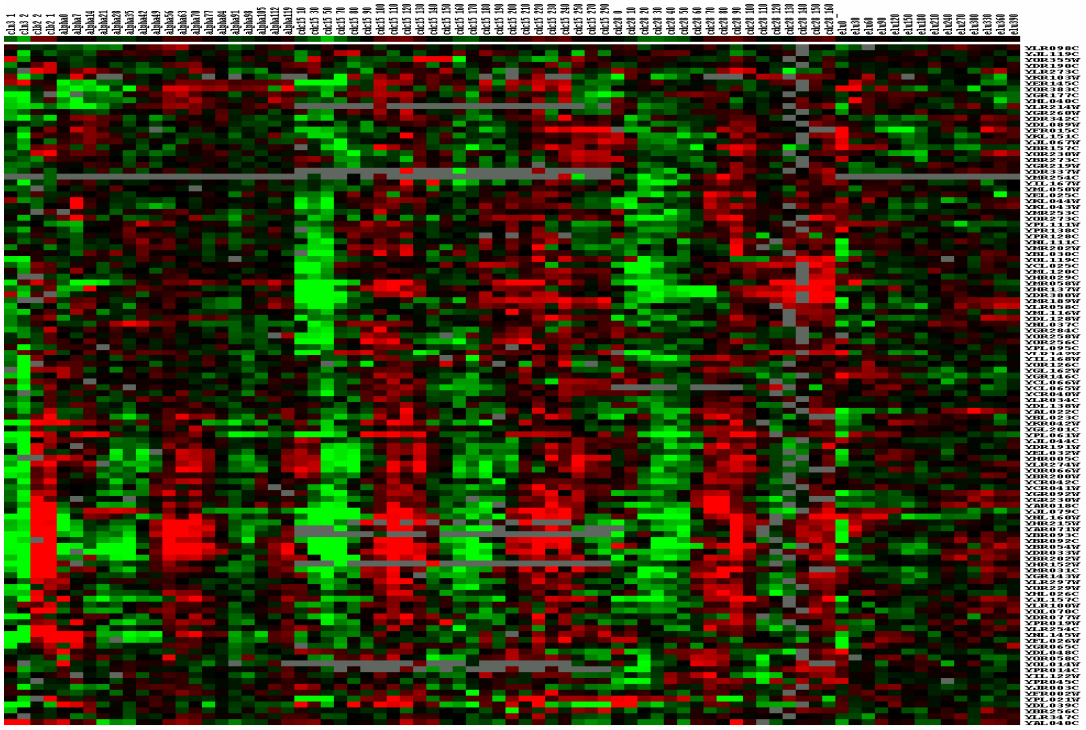

Fig. 4.5 MCM re-clustered cluster (116 genes).

For demonstration, Table 4.8 below shows some interesting associations discovered from each cluster represented in easily understandable if-then rule format.

TABLE 4.8
INTERESTING ASSOCIATIONS DISCOVERED FROM EACH
RE-CLUSTERED CLUSTER IN IF-THEN RULE REPRESENTATION
(GENE EXPRESSION DATA)

If cdc15_70 = [0.32, 1.01] then CLB2 [0.76]	If cdc28_30 = [0.32, 1.01] then MET [0.74]
If cdc15_140 = [-2.56, -0.2] then Histone [0.81]	If cln3_1 = [1.01, 3.09] then CLN2 [0.84]
If elu330 = [0.32, 1.01] then SIC1 [0.82]	If alpha77 = [-2.56, -0.2] then MET [0.80]
If alpha21 = [1.01, 3.09] then CLN2 [0.76]	If cdc15_100 = [-2.56, -0.2] then MAT [0.72]
If cln3_1 = [0.32, 1.01] then Y' [0.80]	If elu150 = [0.32, 1.01] then Histone [0.72]
If alpha56 = [-2.56, -0.2] then MAT [0.78]	If cdc28_70 = [1.01, 3.09] then CLB2 [0.75]
If clb2_1 = [1.01, 3.09] then CLB2 [0.82]	If alpha21 = [0.32, 1.01] then Y' [0.82]
If cdc28_20 = [1.01, 3.09] then CLN2 [0.77]	If alpha_14 = [0.32, 1.01] then MCM [0.79]
If alpha56 = [1.01, 3.09] then MCM [0.80]	If cdc15_30 = [1.01, 3.09] then SIC1 [0.72]
If cdc28_70 = [-0.2, 1.01] then Y' [0.76]	If cln3_2 = [-0.2, 1.01] then MCM [0.78]

The interpretation of the above rules is as follows. For example, for the discovered rule:

If cdc28_30 = [0.32, 1.01] then MET [0.74].

The rule states that if the expression value of a gene under the experimental condition, cdc28_30, is within the interval [0.32, 1.01], then there is a probability of 0.74 that it belongs to cluster MET. Another example, the discovered rule:

If cdc15_140 = [-2.56, -0.2] then Histone [0.81].

The above rule states that if the expression value of a gene under the experimental condition, cdc15_140, is within the interval [–2.56, –0.2], then there is a probability of 0.81 that it belongs to cluster Histone.

Some of the rules given in Table 4.8 are consistent with the findings presented in [141] and this is an indication that the associations uncovered in each discovered cluster are biologically significant. For example, the authors mentioned that genes in CLN2 cluster were induced by GAL-CLN3, one of the associations that we discovered in this cluster is "If cln3_1 = [1.01, 3.09] then CLN2 [0.84]". Moreover, the discovered associations "If clb2_1 = [1.01, 3.09] then CLB2 [0.82]" is consistent with genes in CLB2 cluster that were induced by GAL-CLB2.

Biological Interpretation:

Other than evaluating the results statistically, we have also evaluated the clustering results according to their biological significance. For evaluation, we used a motif discovery algorithm described in [78] to determine if any binding sites located in the promoter regions (from SGD [19]) of the genes in each cluster can be identified (as discussed in Chapter 2, Section 2.3.3). The biological meanings of the discovered binding sites were then validated based on published literature [19], [79]. Since many regulatory sites can be detected with hexanucleotide analysis [78], we also set the oligonucleotide length to be six. Table 4.9 shows the summary of the known binding sites discovered in each re-clustered cluster.

TABLE 4.9
SUMMARY OF THE DISCOVERED BINDING SITES IN EACH
RE-CLUSTERED CLUSTER

Re-clustered cluster	*Sequence revealed*	*Binding site name*
CLB2	CCAAAG	Mcm1
	GGTCAA (potential variant)	SFF
CLN2	ACGCGT	MCB
	ACGCGA (potential variant)	MCB
Histone	ACCAAG (potential variant)	SCB
	TTCTGG	Mcm1
MAT	GTTTCA (potential variant)	Mcm1
MCM	TCCAAA (potential variant)	Mcm1
MET	CACGTG	Met4/Met28/ Cbf1
SIC1	ACCAGC	Swi5;Ace2
	GCCAGC	Swi5;Ace2
Y’	ATGTGG	Mcm1

DNA microarray technology is becoming increasingly important in the analysis of bio-molecules. They provide information that may lead to the understanding of the mechanisms that control gene expression at the transcription level. Because of the large amount of expression data collected everyday and due to the very noisy nature in the data collection process, interpreting and comprehending the experimental results has become a big challenge. Therefore, an effective data mining technique that is also easily interpretable is required.

In this chapter, we have proposed a two-phase clustering algorithm that uses a two-phase approach to clustering gene expression data. The proposed algorithm is able to utilize both local and global information by computing both a local pair-wise distance between two gene expression profiles in Phase 1 and a global probabilistic measure of interestingness of associations in Phase 2. And also, it is able to distinguish between relevant and irrelevant expression levels when performing re-

clustering and make explicit the associations discovered in each cluster for possible interpretation.

The experimental results show that the proposed algorithm can be an effective method for discovering clusters in the presence of noisy data. It is able to assign genes, whose cluster memberships cannot be easily determined by existing clustering methods, into the appropriate clusters. When identifying regulatory motifs at the promoter regions of the co-expressed genes in the discovered clusters, some known binding sites can be discovered. These binding sites can provide explanations for the co-expressed patterns. In addition, the discovered interesting associations, which specify the expression levels under a particular set of experimental conditions the genes should have in each cluster, may lead to further understanding of the mechanism of gene expression.

Chapter 5

Clustering of Gene Expression Data Using Evolutionary Computation

Since external knowledge is seldom available for gene expression data, and also, for a clustering algorithm to discover the best data grouping, it has to consider

$$n(N, k) = \frac{1}{k!}\sum_{i=0}^{k}(-1)^{i}\binom{k}{i}(k-i)^{N} \tag{5.1}$$

possibilities, where N is the total number of records, and k is the total number of clusters [97]. To find the optimal grouping among the very large number of possibilities, there is a need to have an effective clustering algorithm.

Evolutionary algorithms (EAs) have been successfully used to solve different data mining problems [39]-[40], [105]. They have, in particular, been used for clustering [21], [41], [59], [80], [87], [106]-[107], [128]. In [41] and [106], for example, the data records are encoded as *genes*[2] in a *chromosome* and are given a label from one to k, where k is the maximum number of clusters to be discovered. Such algorithms are relatively easy to implement as they do not require special evolutionary operators. Unfortunately, they are not very scalable. As the length of each *chromosome* is exactly the size of the training set, these algorithms are not very practical when handling large data sets.

[2] The terms such as *chromosomes* and *genes*, when used in a computational context, may not have the same meanings as their biological counterparts. In order to avoid possible confusion, when referring to these terms in the contexts of evolutionary computation, they are made italic.

An alternative data and cluster representation was proposed in [128] where the clustering problem is formulated as a graph-partitioning problem. Based on it, each data record is represented as a node in a graph and each node is mapped to a certain position in a *chromosome* and is encoded as a *gene*. The indices of other records are encoded as *alleles* so that if a *gene i* contains value *j*, an edge is created in the graph to link the nodes *i* and *j*. The *alleles* in each *gene i* are therefore the nearest neighbors of *i*, and the users are required to specify the number of nearest neighbors as an input parameter ahead of time. With this representation, an evolutionary algorithm is used to find clusters which are represented as connected sub-graphs. This approach is again not very scalable. Other than the length of the *chromosome*s being again the same as the size of a data set, there is an additional need for the nearest neighbors of data records to be computed. It also suffers from the same problems as other clustering algorithms that are based on the need to compute pair-wise distance measures.

One other popular use of evolutionary algorithms in clustering is to use them to identify the best cluster centers. In [80], [107], each *chromosome* encodes the coordinates of *k* centers and the standard genetic algorithm (GA) is used to find the best ones. A similar approach to identifying the best centers is to use an EA to search for optimal initial seed values for cluster centroids [21]. As in other problems, in clustering we can use domain knowledge in several ways to try to improve the performance of the algorithm. For example, we could design specialized evolutionary operators or we can hybridize the evolutionary algorithm with a conventional clustering algorithm such as the *k*-means algorithm. In [59], [87], each *chromosome* represents the coordinates of the cluster centroids and different crossover methods are used to generate the offspring. After crossover each

chromosome undergoes several iterations of the *k*-means clustering algorithm. The authors observed that adding the *k*-means iterations is crucial for obtaining good results, and although there can be a considerable increase of the computation time if many iterations are used. This kind of hybridization raises the question of how to allocate the computing time. For example, using many generations of the EAs and a few iterations of the local methods or running the EAs for a few generations and using the local methods to improve the solutions. In principle, the centroid-based representation has the advantage that the *chromosomes* are shorter because they only need to encode the coordinates of the *k* centroids. This means that the length of the *chromosome* is proportional to the dimensionality of the problem and not the size of the training set. However, just like many EA-based clustering methods, the drawback of the centroid-based representation is that the number of clusters needed to be specified in advance. Moreover, the similarity functions used such as Euclidean distance or correlation coefficient for measuring the similarity of the records do not differentiate between the importance of different attributes. Therefore, they do not give accurate measurements when the data concerned are noisy and contain missing values. In addition, these similarity functions measure only pair-wise distances, the measurements obtained could be too local.

Clustering gene expression data as a new area of research poses new challenges due to its unique data nature that the previous EA-based clustering algorithms were not originally designed to deal with. As discussed before, there are some new challenges in dealing with gene expression data. For example, the presence of both biological and technical noise inherent in the data. And also, the clustering structure of gene expression data is usually unknown. To effectively tackle the challenges posed by gene expression data, we propose an effective *evo*lutionary *cluster*ing

algorithm called EvoCluster [111], [119], [121]. Compared with other evolutionary and non-evolutionary based clustering algorithms (as discussed in Chapter 2, Section 2.3.3), EvoCluster has several desirable characteristics. It encodes the entire cluster grouping in a *chromosome* so that each *gene* encodes one cluster and each cluster contains the labels of the data records grouped into it. Then, given the above encoding scheme, it has a set of special crossover and mutation operators that facilitates the exchange of grouping information between two *chromosomes* on one hand and allows variation to be introduced to avoid trapping at local optima on the other. And also, unlike many similarity measures that are based on local pair-wise distances [84] that may not give very accurate measurements in the presence of very noisy data, the fitness measure (the association-discovery technique discussed in Chapter 3, Section 3.1) used is probabilistic and it takes into consideration global information contained in a particular grouping of data. It is able to distinguish between relevant and irrelevant feature values (expression levels) in the data during the clustering process, and explain clustering results by explicitly revealing hidden associations discovered in each cluster. In addition, there is no requirement for the number of clusters to be decided in advance. In the following section, the details of EvoCluster will be given.

5.1 EvoCluster: An Evolutionary Clustering Algorithm

To describe EvoCluster [111], [119], [121], let us assume that we are given a set of gene expression data, *G*, consisting of the data collected from *N* genes in *M* experiments carried out under different experimental conditions. Let us represent the data set as a set of *N* genes (records), $\mathbf{G} = \{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_i, \ldots, \boldsymbol{g}_N\}$, with each gene, $\boldsymbol{g}_i$,

$i = 1,..., N$, characterized by M different experimental conditions, $E_1,\ldots,E_j,\ldots,E_M$, whose values, $e_{i1},\ldots,e_{ij},\ldots,e_{iM}$, where e_{ij} represents the expression value of the i^{th} gene under the j^{th} experimental condition (Fig. 4.1).

Like other evolutionary algorithms [22]-[24], [60], [64], [108], EvoCluster consists of the following steps:

1. Initialize a population of *chromosomes* with each representing a unique cluster grouping.
2. Evaluate the fitness of each *chromosome*.
3. Select *chromosomes* for reproduction using the roulette wheel selection scheme.
4. Apply crossover and mutation operators.
5. Replace the least-fit *chromosomes* in the existing population by the newly generated offspring.
6. Repeat Steps 2 to 5 until the stopping criteria are met.

A. Cluster Encoding in Chromosomes and Population Initialization

To evolve the best cluster grouping, EvoCluster encodes different grouping arrangements in different *chromosomes* so that one *chromosome* encodes one particular cluster grouping [55]-[56], [60]. In each such *chromosome*, each *gene* encodes one cluster. Hence, if a particular *chromosome* encodes *k* clusters, $C_1,\cdots,C_i,\cdots,C_k$, it has *k genes*. Since each cluster contains a number of data records, a *gene* encoding a cluster can be considered as being made up of the labels of a number of data records in gene expression data. For example, assume that C_i contains n_i records, $\boldsymbol{g}_{(i1)},\cdots,\boldsymbol{g}_{(ij)},\cdots,\boldsymbol{g}_{(in_i)}$, where $\boldsymbol{g}_{(ij)} \in \mathrm{G} = \{\boldsymbol{g}_1,\ldots,\boldsymbol{g}_i,\ldots,\boldsymbol{g}_N\}$, the

labels of these records can be encoded in each *gene*, C_i so that, a *chromosome* that encodes a particular cluster grouping can then be represented diagrammatically as shown below (Fig. 5.1).

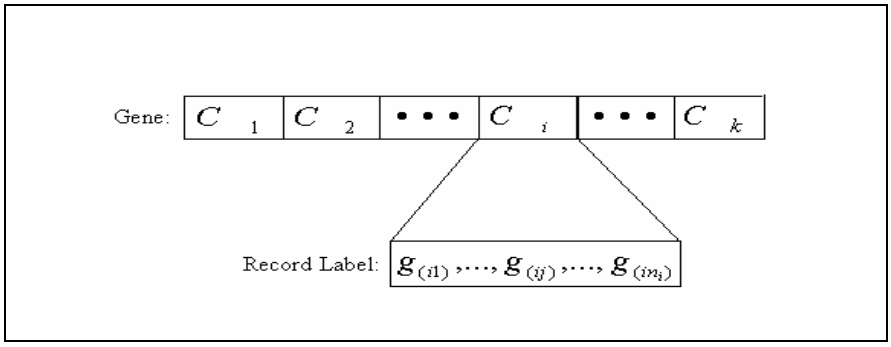

Fig. 5.1 The chromosome encoding scheme.

For the initial population, each *chromosome* is randomly generated in such a way that the number of clusters, *k*, to be encoded in a *chromosome* is first generated randomly from within a certain range of acceptable numbers. Each of the records in $\mathbf{G}=\{\boldsymbol{g}_1,..,\boldsymbol{g}_i,..,\boldsymbol{g}_N\}$ is then assigned, also randomly, to one of the *k* clusters.

B. Selection and Reproduction

Reproduction in EvoCluster consists of the application of both the crossover and mutation operations. As the evolutionary process enters into reproduction, two *chromosomes* are selected as parents for crossover using the roulette-wheel selection scheme [108] so that each parent's chance of being selected is directly proportional to its fitness.

Since it is the cluster grouping encoded in each *chromosome* that conveys the most important information, our crossover operators are designed to facilitate the exchange of grouping information. And since this process can be "guided" or "unguided", our crossover operators are also classified in the same way. We have a "guided" operator and an "unguided" operator. For the "guided" crossover (GC)

operator, the exchange of grouping information is not totally random in the sense that the grouping information of the "best-formed" clusters is preserved during the crossover process. For the "unguided" crossover (UGC) operator, the exchange of the grouping information between clusters takes place randomly.

Assume that two parent *chromosomes*, *P1* and *P2* are chosen so that *P1* encodes k_1 *genes*, $C_1^{P1},\cdots,C_i^{P1},\cdots,C_{k_1}^{P1}$ (with each corresponding to a cluster), and *P2* encodes k_2 *genes*, $C_1^{P2},\cdots,C_i^{P2},\cdots,C_{k_2}^{P2}$, i.e., the number of clusters encoded in each *chromosome* can be different. Assume also that *MIN* is a user-defined minimum number of clusters encoded in a *chromosome* and *MAX* is a user-defined maximum number of clusters encoded in a *chromosome*, then the following are the steps taken by the guided and unguided operators when crossover is performed. It should be noted that the probability for a *gene* or a record label in a *gene* to be selected by both crossover and mutation operators can be randomly generated from within a certain range $[L_g, U_g]$ or $[L_r, U_r]$ respectively, where $0.0 \le L_g \le U_g \le 1.0$ and $0.0 \le L_r \le U_r \le 1.0$, and L_g, U_g, L_r and U_r can be set by users or also generated randomly. Moreover, in both guided crossover and guided mutation operators, the interestingness of each *gene* is determined based on the fitness measure described in part C below.

The Guided Crossover (GC) and Unguided Crossover (UGC) Operators

1. Set $P_{\text{g-rpl}}$, the probability for a *gene* to be selected for crossover.
2. Set $P_{\text{r-rpl}}$, the probability for a record label in a *gene* to be replaced by another record label from another *gene* in another parent.
3. *Gene* selection procedure:

a. For the UGC, based on $P_{\text{g-rpl}}$ and using a random number generator, each *gene* in *P1* is scanned to decide if it should be selected. Those selected are then represented as $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$ where $l_1 < k_1$, k_2 and $C_{(i)}^{P1}$, $i=1,...,l_1$, is in $\{C_1^{P1},\cdots,C_i^{P1},\cdots C_{k_1}^{P1}\}$.

b. For the GC, based on $P_{\text{g-rpl}}$, $N_{\text{g-rpl}} < k_1, k_2$, the number of interesting *genes* to be selected, can be determined. Then select $N_{\text{g-rpl}}$ of the most interesting ones in *P1* and *P2* respectively. Rank them in descending order of interestingness and represent them as $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$ and $\{C_{(1)}^{P2},\cdots,C_{(i)}^{P2},\cdots,C_{(l_1)}^{P2}\}$ so that $C_{(1)}^{P1}$ and $C_{(1)}^{P2}$ is the most interesting in *P1* and *P2* respectively.

4. Record label replacement procedure:

a. For the UGC, for each *gene* in $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$, say $C_{(i)}^{P1}$, randomly select one *gene* from *P2*, say C_i^{P2} that has not previously been selected. Based on $P_{\text{r-rpl}}$ and using a random number generator, each record label in $C_{(i)}^{P1}$ can be scanned to identify those that should be replaced by a record label in C_i^{P2}. Record labels that are selected for replacement are then represented as $\{g_{(i1)}^{P1},\ldots,g_{(ij)}^{P1},\ldots,g_{(in_i)}^{P1}\}$ and removed from $C_{(i)}^{P1}$. Randomly select n_i or $|C_i^{P2}|$ record labels, whichever is smaller, from C_i^{P2} and replace those removed. Repeat the above steps for all the other selected *genes* in *P1*.

b. For the GC, begin with the most interesting *gene*, say $C_{(1)}^{P1}$, select the corresponding most interesting *gene*, $C_{(1)}^{P2}$, from *P2*. Based on $P_{\text{r-rpl}}$, scan

through each record label in $C_{(1)}^{P1}$ to select those that should be replaced by that in $C_{(1)}^{P2}$. Those record labels to be replaced are represented as $\{g_{(11)}^{P1},\ldots,g_{(1j)}^{P1},\ldots,g_{(1n_1)}^{P1}\}$ and removed from $C_{(1)}^{P1}$. Randomly select n_i or $|C_{(1)}^{P2}|$ record labels, whichever is smaller, from $C_{(1)}^{P2}$ and replace those removed. Repeat the above steps for all the other selected *genes* in *P1*.

5. Repairing procedure (for producing child *Ch1*): Scan through all *genes* to remove duplicates in such a way that if a record label is found in another *gene*, other than the one containing the replacement, it is removed. For those record labels that have not been assigned to any *genes* after their removals:
   a. For the UGC, they are randomly assigned to one of the *genes*.
   b. For the GC, EvoCluster constructs a classifier based on *Ch1* using a re-classification algorithm described in [34]. They are then re-classified into one of the *genes* encoded in *Ch1*.
6. Repeat Steps 1-5 with *P2* to produce child *Ch2*.

After crossover, the children produced undergo mutation in order to avoid getting trapped at local optima on one hand and to ensure diversity on the other. EvoCluster makes available six different mutation operators that can be selected at random when a *chromosome* undergoes mutation. These operators can be classified according to whether or not the mutation process involves just the removal and reclassification of record labels or the merging and splitting of the whole *gene*. They can also be classified according to whether or not they are “guided” or “unguided”. Based on these classification schemes, EvoCluster makes use of six operators as follows:

(a) The unguided remove-and-reclassify-record mutation (UGRRM) operator.

(b) The guided remove-and-reclassify-record mutation (GRRM) operator.

(c) The unguided merge-*gene* mutation (UGMGM) operator.

(d) The guided merge-*gene* mutation (GMGM) operator.

(e) The unguided split-*gene* mutation (UGSGM) operator.

(f) The guided split-*gene* mutation (GSGM) operator.

The merge and split mutation operators (i.e., UGMGM, GMGM, UGSGM, and GSGM) are specifically designed to allow the length of *chromosomes* to be changed dynamically as the evolutionary process progresses. The advantage with this feature is that the number of clusters that need to be formed does not need to be specified by the users ahead of time. In the following, the details of these operators are given:

The Guided (GRRM) and Unguided Remove-and-Reclassify-Record Mutation (UGRRM) Operators

1. Set $P_{\text{g-rr}}$, the probability for a *gene* to be selected.
2. Set $P_{\text{r-rr,}}$ the probability for a record label in a *gene* to be removed.
3. *Gene* selection procedure:
   a. For the UGRRM, based on $P_{\text{g-rr}}$, scan through each *gene* in the *chromosome* to decide if it should be selected. Those selected are then represented as $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$ where $l_1 < k_1$ and $C_{(i)}^{P1}$, $i = 1,...,l_1$, is a member of $\{C_1^{P1},\cdots,C_i^{P1},\cdots C_{k_1}^{P1}\}$.
   b. For the GRRM, based on $P_{\text{g-rr}}$, determine $N_{\text{g-rr}} < k_1$, the number of uninteresting *genes* to be selected. Then select the $N_{\text{g-rr}}$ least interesting *genes* and represent them as $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$ where $l_1 < k_1$ and $C_{(i)}^{P1}$,

$i = 1,...,l_1$, is a member of $\{C_1^{P1},\cdots,C_i^{P1},\cdots C_{k_1}^{P1}\}$.

4. Record label replacement procedure: For each *gene* in $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$, based on $P_{\text{r-rr}}$, scan through each record label in each of $C_{(i)}^{P1}$ to select those that should be removed. These record labels are represented as $\{g_{(i1)}^{P1},\ldots,g_{(ij)}^{P1},\ldots,g_{(in_i)}^{P1}\}$ and removed from $C_{(i)}^{P1}$.
5. Children repairing procedure: For those record labels that have not been assigned to any *genes* after their removals:
   a. For the UGRRM, they are randomly classified into one of the *genes.*
   b. For the GRRM, EvoCluster constructs a classifier based on the child *chromosome* using a re-classification algorithm described in [34]. They are then re-classified into one of the *genes* encoded in the *chromosome*.

The Guided (GMGM) and Unguided Merge-*Gene* Mutation (UGMGM) Operators

1. Set $P_{\text{g-mrg}}$, the probability for a *gene* to be merged.
2. *Gene* selection procedure:
   a) For the UGMGM, based on $P_{\text{g-mrg,}}$ scan through each *gene* in the *chromosome* to select a *gene* for merging. Those selected are then represented as $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$ where $l_1 < k_1$ and $C_{(i)}^{P1}$, $i = 1,...,l_1$, is a member of $\{C_1^{P1},\cdots,C_i^{P1},\cdots,C_{k_1}^{P1}\}$.
   b) For the GMGM, based on $P_{\text{g-mrg}}$, determine $N_{\text{g-mrg}} < k_1$, the number of uninteresting *genes* to be merged. Then select $N_{\text{g-mrg}}$ least interesting *genes* and represent them in ascending order of interestingness as $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$ so that $C_{(1)}^{P1}$ is the least interesting, where $l_1 < k_1$ and

$C_{(i)}^{P1}$, $i = 1,...,l_1$, is a member of $\{C_1^{P1},\cdots,C_i^{P1},\cdots C_{k_1}^{P1}\}$.

3. Merging procedure: For each *gene* in $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$, randomly select one other *gene* to be merged in this set. The number of *genes* remaining after merging should be greater than *MIN*. Otherwise, the mutation operator terminates.

The Guided (GSGM) and Unguided Split-*Gene* Mutation (UGSGM) Operators

1. Set $P_{\text{g-splt}}$, the probability for a *gene* to be split.
2. *Gene* selection procedure:
   a) For the UGSGM. Based on $P_{\text{g-splt}}$, scan through each *gene* in the *chromosome* to decide if it should be selected for splitting. Those selected are then represented as $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$ where $l_1 < k_1$ and $C_{(i)}^{P1}$, $i = 1,...,l_1$, is a member of $\{C_1^{P1},\cdots,C_i^{P1},\cdots C_{k_1}^{P1}\}$.
   b) For the GSGM. Based on $P_{\text{g-splt}}$, determine $N_{\text{g-splt}} < k_1$ the number of uninteresting *genes* to be split. Then select $N_{\text{g-splt}}$ least interesting *genes*, and represent them in ascending order of interestingness as $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$ so that $C_{(1)}^{P1}$ is the least interesting, where $l_1 < k_1$ and $C_{(i)}^{P1}$, $i = 1,...,l_1$, is a member of $\{C_1^{P1},\cdots,C_i^{P1},\cdots C_{k_1}^{P1}\}$.
3. Splitting procedure: For each *gene* in $\{C_{(1)}^{P1},\cdots,C_{(i)}^{P1},\cdots,C_{(l_1)}^{P1}\}$, randomly split it into two clusters. The resulting number of *genes* has to be smaller than *MAX*. Otherwise, the mutation operator terminates.

A simple evolutionary algorithm typically uses a generational replacement technique. This technique replaces the entire population after enough children are generated. However, the potential drawback of such a replacement approach is that many good *chromosomes* also get replaced, making it difficult for the good traits to survive. To overcome this problem, EvoCluster adopts a steady state reproduction approach [23] so that only two least-fit *chromosomes* are replaced whenever two new children are generated after each reproduction.

C. Fitness Function

To evaluate the fitness of each *chromosome*, EvoCluster adopt the association-discovery technique (as discussed in Chapter 3, Section 3.1) as an objective fitness measure. This method has the advantage that it is able to handle the potential noise resulting from the clustering process. Since, this technique is able to take into consideration the global information by distinguishing between the expression levels that are relevant and irrelevant in a particular cluster grouping. This makes EvoCluster very robust even in the presence of noisy data.

The fitness evaluation procedure is invoked after new *chromosomes* are formed. The fitness function accepts a *chromosome* as a parameter and its fitness is evaluated in two steps (the details of these two steps are described in Chapter 4, Section 4.1, phase 2). In Step one, it attempts to discover interesting associations in the cluster grouping encoded in the *chromosome*. To do so, a subset of records (for application here, 70% of records) from different clusters encoded in a *chromosome* is selected randomly to form a data set for training. In Step two, those records were not selected in Step one are re-classified into one of the clusters based on the discovered associations. Then, the predicted label can be compared with the original label of

each record encoded in the *chromosome* to determine the re-classification accuracy and based on it, the fitness value of the cluster grouping encoded in a *chromosome* can be determined. As discussed before, if a clustering algorithm is effective, the discovered clusters should contain patterns that can be used to accurately re-classify the records in the testing data. And if this is the case, the re-classification accuracy measure is an indication of how good the quality of the cluster grouping is. For this reason, the re-classification accuracy is then taken to be the fitness of each *chromosome*.

5.2 Experiments

A. Experimental Data

For experimentation, we used a set of simulated data consisting of 300 records each characterized by 50 different attributes that takes on values from [0.0, 1.0]. Initially, all these records were sampled from a uniform distribution and they were pre-classified into one of three clusters so that each cluster contains 100 records. To embed hidden patterns in the data, 10% of the attributes in each cluster were randomly selected. For each selected attribute, 30%-40% of its values in that cluster were randomly generated from within a certain range [L, U], where $0.0 \leq L \leq U \leq 1.0$ so that L was selected uniformly from [0.0, 1.0] first, and U was then also selected uniformly from [L, 1.0].

In addition to the simulated data, to test the effectiveness of EvoCluster, we also used two different sets of real expression data. For *Dataset 1*, it contains a subset of about 380 genes measured under 17 different experimental conditions and we tried in our experiments to partition this data set into different clusters from 4 to 8 [29].

For *Dataset* 2, it contains a subset of about 800 genes measured under 77 different experimental conditions and we also tried to partition this data set into different clusters from 6 to 10 [141]. For performance evaluation, EvoCluster was evaluated based on three objective measures as discussed in Chapter 4, Section 4.2. They are the F-measure, the predictive power measure, and the Davies-Bouldin validity index (DBI) measure.

5.3 Results and Discussions

The effectiveness of EvoCluster has been compared with a number of different clustering algorithms (as discussed in Chapter 2, Section 2.3.3) using both simulated and real data.

In our experiments, we adopted the default settings for the parameters of SOM [85] as described in [149] (i.e., the bubble neighborhood function, the initial learning weight ($Alpha\_i$) was set to 0.1, the final learning weight ($Alpha\_f$) was set to 0.005, the initial sigma ($Sigma\_i$) was set to 5 and the finial sigma ($Sigma\_f$) was set to 0.5, etc.). Since a given number of clusters (say 6) can represent multiple SOM geometries (i.e., 1x6, 2x3, etc), we also tried all these geometries in order to obtain the best cluster grouping with SOM. For both SOM and the *k*-means algorithms [100], 5000 iterations were performed. Also, to ensure that the best results for them were obtained, 100 runs were performed for each of them with each run using different randomly-generated initial cluster centroids. Only the best result from among these 100 runs was recorded. Afterward, such 100-run test was repeated 10 times. The 10 best results obtained from each 100-run test were then recorded (Table 5.1).

In the case of EvoCluster, we also performed 10 trials in our experiments. For each such trial, we randomly generated different initial populations of size fixed at 50. Using a steady-state reproduction scheme, the evolutionary process was terminated either when the maximum *chromosome* fitness converged or when the maximum number of reproductions reached 5000. As summarized in Table 5.1, the total number of iterations performed with *k*-means and SOM is 100 times more than the total number of reproductions carried out using EvoCluster. This was done to ensure that EvoCluster would not use any more computational resources (in terms of the number of trial-and-errors through iterations/reproductions) than other clustering algorithms it was being compared against.

TABLE 5.1
SUMMARY OF THE NUMBER OF REPRODUCTIONS/ITERATIONS PERFORMED BY EVOCLUSTER, *K*-MEANS AND SOM

	No. of reproductions /iterations	*No. of runs*	*No. of trials*
EvoCluster	5000	-	10
k-means	5000	100	10
SOM	5000	100	10

During the evolutionary process, the probabilities of selection of a *gene* or a record label in a *gene*, used by the crossover and mutation operators, were randomly generated from within [0.2, 0.8] using a random number generator [108].

In order to evaluate the effectiveness of EvoCluster, in addition to the traditional clustering algorithms, we also compared its performance with a clustering algorithm that represents one of the most successful attempts to use EA in clustering [107]. Each *gene* in it encodes one dimension of a cluster center and a *chromosome* encodes a fixed number of clusters. For our experiment with it, we set the crossover rate to 0.8 and the mutation rate to 0.001, i.e., the same as that used in [107]. Other parameter settings, including population size, number of reproductions, etc., were set

exactly the same as that with EvoCluster.

Since one of the desirable features of EvoCluster is its ability to distinguish relevant from irrelevant feature values during the evolutionary process, we also compared its performance against various hybrid clustering algorithms that use a feature selection technique in combination with a clustering algorithm. Specifically, we used an effective feature selection technique together with the *k*-means algorithm, SOM, the hierarchical clustering algorithm, and the EA-based algorithm to see how much improvement the performances of these algorithms can have when features were first filtered for clustering. Among different feature selection techniques that can be used for this purpose [49], [65], [143], [146], [160], we chose to consider the one described in [143], [146]. This is because this technique, which makes use of the *t*-statistic measure, has been popularly used to reduce the number of attributes in gene expression data. Given an initial cluster grouping, the feature selection was adopted and performed in several steps as follows:

(i) A cluster grouping is first determined using say, the *k*-means algorithm (or SOM, or the hierarchical clustering algorithm, or the EA-based algorithm [107], etc.) that the feature filtering method is hybridizing with.

(ii) Given the initial cluster grouping, a *t*-statistic measure is then computed for each attribute to determine how well it is able to distinguish one cluster from the rest of the others.

(iii) Based on the *t*-statistic, a new subset of attributes with the largest *t*-values is obtained by first selecting 5% of the attributes that has the largest *t*-values. With this new attribute subset, a classifier is then generated using C4.5 [131] and its classification accuracy is measured using ten-fold cross validation. Afterward, the process of adding another 5% of the attributes with the largest

t-values to this new attribute subset and measuring the accuracy of the resulting new classifier is repeated. The final attribute subset is determined when the performance of the classifier converge [143].

(iv) With this final attribute subset, a new and improved cluster grouping is then determined.

For performance evaluation, we also compared the performance of EvoCluster with the two-phase clustering algorithm proposed in Chapter 4. By using the *k*-means algorithm, SOM, the hierarchical clustering algorithm and the EA-based clustering algorithm in the first phase respectively and then applying the re-clustering phase in the second, the re-clustered result of each algorithm can be obtained.

1. Simulated Data

Since the number of clusters (*k*=3) to discover was known-in-advance for the simulated data, the length of the *chromosome* was fixed in our experiment to be 3 (the merge and split mutation operators were not used in the simulated data). Table 5.2 shows the parameter settings of EvoCluster used in the simulated data.

TABLE 5.2
PARAMETER SETTINGS OF EVOCLUSTER USED IN SIMULATED DATA
(P_c REPRESENTS THE PROBABILITY OF GUIDED OR UNGUIDED CROSSOVER OPERATOR SELECTED AND P_m REPRESENTS THE PROBABILITY OF GUIDED OR UNGUIDED MUTATION OPERATOR SELECTED)

	Pop. Size	*MIN*	*MAX*	P_c	P_m	L_g	U_g	L_r	U_r
Simulated data	50	3	3	0.5	0.5	0.2	0.8	0.2	0.8

As discussed in the previous section, EvoCluster has a set of “guided” and “unguided” operators. For the “guided” operators, the exchange of grouping

information is not totally random in the sense that the grouping information of the "best-formed" clusters is preserved during the evolutionary process. For the "unguided" operators, the exchange of the grouping information between clusters takes place randomly. To determine if there is a real need for both types of operators, three separate experiments were carried out on the simulated data. In the first experiments, a 50-50 mixture of "guided" or "unguided" operators were used whereas in the second and third, only "guided" operators and only "unguided" operators were used respectively. The average number of reproductions performed by each algorithm until convergence and also the average F-measure are given in Table 5.3.

TABLE 5.3
COMPARISON OF THE CLUSTERING PERFORMANCES OBTAINED BY USING "GUIDED+UNGUIDED" OPERATORS, "GUIDED" OPERATORS, OR "UNGUIDED" OPERATORS
(SIMULATED DATA)

	k	*No. of reproduction (convergence)*	*F-measure*
Unguided + Guided operators	3	3830	0.91
Unguided operators	3	4561	0.63
Guided operators	3	3050	0.80

As shown in Table 5.3 and as expected, when only "guided" operators were used alone, it appeared that the result converged only to some local optima and when only "unguided" operators were used alone, not only a longer evolutionary process was required, the result obtained was unsatisfactory. The performance of EvoCluster is at its best when both "guided" and "unguided" operators were used together even though it required more reproductions to converge (compared to "guided" operators alone). Based on these results, we conclude that both the "guided" and "unguided"

operators have a role to play in the evolutionary process. When they are used together, they can facilitate the exchange of grouping information in a way that such information in the "best-formed" clusters is preserved as much as possible during the evolutionary process on one hand but variations can be introduced at the same time on the other so as to avoid trapping at local optima too early. The performance of EvoCluster has been compared with other clustering algorithms and the results are given in Table 5.4 below.

TABLE 5.4
COMPARISON OF THE AVERAGE F-MEASURE
(SIMULATED DATA)

	EvoCluster	*EA-based*	*EA-based + FS*	*Re-clustered EA-based*	*k-means*
k=3	0.91	0.66	0.78	0.82	0.61
	k-means + FS	*Re-clustered k-means*	*SOM*	*SOM + FS*	*Re-clustered SOM*
k=3	0.75	0.77	0.57	0.72	0.72
	Hierarchical	*Hierarchical + FS*	*Re-clustered Hierarchical*		
k=3	0.52	0.65	0.70		

As shown in the above tables, compared with other clustering algorithms, EvoCluster performs better in terms of the F-measure. Moreover, it seems that none of the EA-based, *k*-means, SOM and hierarchical algorithms (with or without feature selection) is particularly effective when handling very noisy data such as the simulated data. In order to decide if the differences between these clustering algorithms are significantly different, we performed one-sided pair-wise *t*-test [50] on the null and alternative hypotheses of H_0: $\mu_1 > \mu_2$ and H_a: $\mu_1 \leq \mu_2$, respectively. The results of the *t*-tests confirm that the differences are all statistically significant at the 95% confidence level (in Table 5.5). This shows that EvoCluster is very robust in the presence of a very noisy environment.

TABLE 5.5
RESULTS OF *T*-TEST (10 TRIALS)
(SIMULATED DATA) (F REPRESENTS F-MEASURE)

Test #	Measure	Null Hypothesis (H_0: $\mu_1 > \mu_2$)	*t*-test	Accept/Reject
1	F	$\mu_{\text{EvoCluster}} > \mu_{\text{EA-based}}$	+25.23	Accept
2	F	$\mu_{\text{EvoCluster}} > \mu_{k\text{-means}}$	+34.17	Accept
3	F	$\mu_{\text{EvoCluster}} > \mu_{\text{SOM}}$	+42.54	Accept
4	F	$\mu_{\text{EvoCluster}} > \mu_{\text{Hierarchical}}$	+76.22	Accept
5	F	$\mu_{\text{EvoCluster}} > \mu_{\text{EA-based+FS}}$	+19.45	Accept
6	F	$\mu_{\text{EvoCluster}} > \mu_{k\text{-means+FS}}$	+22.36	Accept
7	F	$\mu_{\text{EvoCluster}} > \mu_{\text{SOM+FS}}$	+25.08	Accept
8	F	$\mu_{\text{EvoCluster}} > \mu_{\text{Hierarchical+FS}}$	+33.95	Accept
9	F	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered EA-based}}$	+15.26	Accept
10	F	$\mu_{\text{EvoCluster}} > \mu_{\text{Reclustered } k\text{-means}}$	+19.94	Accept
11	F	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered SOM}}$	+23.86	Accept
12	F	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered Hierarchical}}$	+27.65	Accept

2. Gene Expression Data

The performances of EvoCluster have also been evaluated using real expression data sets. The minimum and maximum number of clusters considered for both *Datasets 1* and *2* were set at (*MIN*=4, *MAX*=8) and (*MIN*=6, *MAX*=10), respectively. Table 5.6 shows the parameter settings of EvoCluster used in *Datasets 1* and *2*.

TABLE 5.6
PARAMETER SETTINGS OF EVOCLUSTER USED
IN GENE EXPRESSION DATA

	Pop. Size	*MIN*	*MAX*	P_c	P_m	L_g	U_g	L_r	U_r
Dataset 1	50	4	8	0.5	0.17	0.2	0.8	0.2	0.8
Dataset 2	50	6	10	0.5	0.17	0.2	0.8	0.2	0.8

As with the simulated data, the experiments with *Datasets 1* and *2* were repeated three times with a mixture of guided and unguided operators, unguided operators alone and guided operators alone respectively. Based on the results showed in Tables 5.7 and 5.8, we found that using both “guided” and “unguided” operators together, once again, gave us the best clustering results.

TABLE 5.7
COMPARISON OF THE CLUSTERING PERFORMANCES OBTAINED BY USING "GUIDED+UNGUIDED" OPERATORS, "GUIDED" OPERATORS, OR "UNGUIDED" OPERATORS
(DATASET 1)

	k	*No. of reproduction (convergence)*	*Predictive power*	*DBI*
Unguided + Guided operators	4	3104	87.10%	1.52
	5	3631	89.86%	1.48
	6	4703	80.90%	1.57
	7	4464	83.42%	1.58
	8	4405	77.33%	1.59
	Average	4061	83.72%	1.55
Unguided operators	4	4852	67.68%	1.62
	5	4197	70.41%	1.61
	6	4763	62.33%	1.65
	7	4945	64.75%	1.64
	8	4572	62.59%	1.65
	Average	4666	65.55%	1.63
Guided operators	4	2670	76.13%	1.57
	5	1978	78.47%	1.57
	6	2404	75.94%	1.59
	7	4204	72.16%	1.60
	8	3978	70.08%	1.61
	Average	3047	74.56%	1.59

TABLE 5.8
COMPARISON OF THE CLUSTERING PERFORMANCES OBTAINED BY USING "GUIDED+UNGUIDED" OPERATORS, "GUIDED" OPERATORS, OR "UNGUIDED" OPERATORS
(DATASET 2)

	k	*No. of reproduction (convergence)*	*Predictive power*	*DBI*
Unguided + Guided operators	6	3325	84.88%	1.59
	7	3701	81.25%	1.64
	8	4088	78.19%	1.67
	9	4717	74.26%	1.67
	10	4779	73.26%	1.68
	Average	4122	78.37%	1.65
Unguided operators	6	4493	67.28%	1.69
	7	4989	65.36%	1.69
	8	4777	57.19%	1.79
	9	4815	58.34%	1.79
	10	4622	59.71%	1.78
	Average	4739	61.58%	1.75
Guided operators	6	2089	73.23%	1.64
	7	3385	71.02%	1.65
	8	2876	67.49%	1.68
	9	4002	66.33%	1.68
	10	4747	68.51%	1.70
	Average	3420	69.32%	1.67

The performances of EvoCluster in comparison with other algorithms are given in Tables 5.9-5.12.

TABLE 5.9
COMPARISON OF THE AVERAGE PREDICTIVE POWER
(DATASET 1)

	EvoCluster	*EA-based*	*EA-based + FS*	*Re-clustered EA-based*	*k-means*
k=4	87.10%	70.23%	77.41%	79.36%	76.43%
k=5	89.86%	73.92%	80.16%	82.27%	70.68%
k=6	80.90%	68.34%	72.77%	74.61%	66.98%
k=7	83.42%	69.12%	75.53%	73.39%	64.77%
k=8	77.33%	60.02%	67.58%	70.48%	63.12%
Avg	83.72%	68.33%	74.69%	76.02%	68.40%
	k-means + FS	*Re-clustered k-means*	*SOM*	*SOM + FS*	*Re-clustered SOM*
k=4	80.02%	81.35%	72.38%	78.21%	76.48%
k=5	77.48%	83.11%	68.53%	75.51%	77.53%
k=6	75.92%	73.09%	64.26%	69.23%	71.47%
k=7	72.37%	71.22%	63.84%	70.06%	72.81%
k=8	69.55%	68.72%	61.39%	70.84%	68.06%
Avg	75.07%	75.50%	66.08%	72.77%	73.27%
	Hierarchical	*Hierarchical + FS*	*Re-clustered Hierarchical*		
k=4	74.39%	75.37%	77.74%		
k=5	75.22%	76.82%	78.18%		
k=6	73.23%	75.16%	77.09%		
k=7	64.66%	66.79%	69.21%		
k=8	62.30%	64.23%	66.11%		
Avg	69.96%	71.67%	74.07%		

TABLE 5.10
COMPARISON OF THE AVERAGE DBI
(DATASET 1)

	EvoCluster	*EA-based*	*EA-based + FS*	*Re-clustered EA-based*	*k-means*
k=4	1.52	1.61	1.57	1.57	1.59
k=5	1.48	1.54	1.51	1.51	1.56
k=6	1.57	1.62	1.60	1.61	1.62
k=7	1.58	1.61	1.60	1.60	1.66
k=8	1.59	1.69	1.65	1.63	1.68
Avg	1.55	1.61	1.58	1.58	1.62
	k-means + FS	*Re-clustered k-means*	*SOM*	*SOM + FS*	*Re-clustered SOM*
k=4	1.56	1.58	1.62	1.58	1.60
k=5	1.52	1.51	1.64	1.55	1.53
k=6	1.60	1.60	1.62	1.59	1.59
k=7	1.61	1.60	1.70	1.67	1.64
k=8	1.64	1.62	1.79	1.72	1.69
Avg	1.59	1.58	1.67	1.62	1.60
	Hierarchical	*Hierarchical + FS*	*Re-clustered Hierarchical*		
k=4	1.61	1.60	1.59		
k=5	1.59	1.56	1.56		
k=6	1.58	1.57	1.57		
k=7	1.68	1.61	1.60		
k=8	1.69	1.67	1.67		
Avg	1.63	1.60	1.59		

TABLE 5.11
COMPARISON OF THE AVERAGE PREDICTIVE POWER
(DATASET 2)

	EvoCluster	*EA-based*	*EA-based + FS*	*Re-clustered EA-based*	*k-means*
k=6	84.88%	72.26%	78.43%	81.37%	67.23%
k=7	81.25%	61.37%	69.55%	73.09%	63.19%
k=8	78.19%	67.12%	72.38%	73.47%	61.40%
k=9	74.26%	60.51%	65.43%	68.81%	64.77%
k=10	73.26%	64.86%	70.24%	71.24%	62.59%
Avg	78.37%	65.22%	71.21%	73.60%	63.84%
	k-means + FS	*Re-clustered k-means*	*SOM*	*SOM + FS*	*Re-clustered SOM*
k=6	75.86%	77.66%	62.88%	69.14%	73.33%
k=7	72.37%	75.91%	63.75%	70.04%	69.42%
k=8	68.88%	70.03%	56.21%	66.56%	69.19%
k=9	70.12%	69.64%	56.87%	68.38%	69.80%
k=10	71.11%	70.98%	48.38%	59.48%	63.75%
Avg	71.67%	72.84%	56.72%	66.72%	69.10%
	Hierarchical	*Hierarchical + FS*	*Re-clustered Hierarchical*		
k=6	65.30%	71.75%	74.78%		
k=7	59.86%	66.32%	64.02%		
k=8	63.11%	68.29%	70.35%		
k=9	62.80%	67.74%	66.67%		
k=10	58.21%	68.86%	69.92%		
Avg	61.85%	68.59%	69.15%		

TABLE 5.12
COMPARISON OF THE AVERAGE DBI
(DATASET 2)

	EvoCluster	*EA-based*	*EA-based + FS*	*Re-clustered EA-based*	*k-means*
k=6	1.59	1.65	1.62	1.61	1.67
k=7	1.64	1.70	1.69	1.67	1.70
k=8	1.67	1.69	1.68	1.68	1.71
k=9	1.67	1.72	1.69	1.69	1.70
k=10	1.68	1.73	1.70	1.70	1.72
Avg	1.65	1.69	1.67	1.67	1.70
	k-means + FS	*Re-clustered k-means*	*SOM*	*SOM + FS*	*Re-clustered SOM*
k=6	1.63	1.62	1.74	1.69	1.68
k=7	1.68	1.67	1.77	1.72	1.70
k=8	1.70	1.69	1.78	1.75	1.72
k=9	1.68	1.68	1.79	1.75	1.75
k=10	1.70	1.70	1.85	1.79	1.75
Avg	1.68	1.67	1.79	1.74	1.72
	Hierarchical	*Hierarchical + FS*	*Re-clustered Hierarchical*		
k=6	1.71	1.70	1.70		
k=7	1.76	1.74	1.72		
k=8	1.73	1.72	1.70		
k=9	1.76	1.75	1.72		
k=10	1.79	1.76	1.75		
Avg	1.75	1.73	1.71		

The F-measure was not computed for the experiments with real data sets as the original correct clustering results are not known. To confirm that these differences are also statistically significant, we performed one-sided pair-wise *z*-test [50] (as the sample size is large enough, the *z*-test is used rather than the *t*-test) on the null and alternative hypotheses of H_0: $\mu_1 > \mu_2$ and H_a: $\mu_1 \leq \mu_2$, respectively, for the case of the predictive power and on the null and alternative hypotheses of H_0: $\mu_1 < \mu_2$ and H_a: $\mu_1 \geq \mu_2$, respectively, for the case of the DBI measure. The results of these tests are shown in Tables 5.13 and 5.14. According to the above tables, EvoCluster again

performs better than others even with the combination of the feature selection method.

TABLE 5.13
RESULTS OF *Z*-TEST (10 TRIALS) (DATASET 1 – OVER ALL *Ks*)
(P REPRESENTS PREDICTIVE POWER AND D REPRESENTS DBI)

Test #	Measure	Null Hypothesis (H_0: $\mu_1 > \mu_2$)	*z*-test	Accept/Reject
1	P	$\mu_{\text{EvoCluster}} > \mu_{\text{EA-based}}$	+14.87	Accept
2	P	$\mu_{\text{EvoCluster}} > \mu_{k\text{-means}}$	+16.13	Accept
3	P	$\mu_{\text{EvoCluster}} > \mu_{\text{SOM}}$	+23.48	Accept
4	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Hierarchical}}$	+21.32	Accept
5	P	$\mu_{\text{EvoCluster}} > \mu_{\text{EA-based+FS}}$	+8.66	Accept
6	P	$\mu_{\text{EvoCluster}} > \mu_{k\text{-means+FS}}$	+8.53	Accept
7	P	$\mu_{\text{EvoCluster}} > \mu_{\text{SOM+FS}}$	+13.36	Accept
8	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Hierarchical+FS}}$	+14.55	Accept
9	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered EA-based}}$	+6.72	Accept
10	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Reclustered } k\text{-means}}$	+7.06	Accept
11	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered SOM}}$	+11.98	Accept
12	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered Hierarchical}}$	+10.33	Accept
Test #	Measure	Null Hypothesis (H_0: $\mu_1 < \mu_2$)	*z*-test	Accept/Reject
1	D	$\mu_{\text{EvoCluster}} < \mu_{\text{EA-based}}$	-11.58	Accept
2	D	$\mu_{\text{EvoCluster}} < \mu_{k\text{-means}}$	-12.26	Accept
3	D	$\mu_{\text{EvoCluster}} < \mu_{\text{SOM}}$	-18.75	Accept
4	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Hierarchical}}$	-17.16	Accept
5	D	$\mu_{\text{EvoCluster}} < \mu_{\text{EA-based+FS}}$	-8.58	Accept
6	D	$\mu_{\text{EvoCluster}} < \mu_{k\text{-means+FS}}$	-9.32	Accept
7	D	$\mu_{\text{EvoCluster}} < \mu_{\text{SOM+FS}}$	-12.17	Accept
8	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Hierarchical+FS}}$	-11.84	Accept
9	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Re-clustered EA-based}}$	-8.21	Accept
10	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Reclustered } k\text{-means}}$	-8.79	Accept
11	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Re-clustered SOM}}$	-11.65	Accept
12	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Re-clustered Hierarchical}}$	-10.37	Accept

TABLE 5.14
RESULTS OF *Z*-TEST (10 TRIALS) (DATASET 2 – OVER ALL *Ks*)
(P REPRESENTS PREDICTIVE POWER AND D REPRESENTS DBI)

Test #	Measure	Null Hypothesis (H_0: $\mu_1 > \mu_2$)	*z*-test	Accept/Reject
1	P	$\mu_{\text{EvoCluster}} > \mu_{\text{EA-based}}$	+13.12	Accept
2	P	$\mu_{\text{EvoCluster}} > \mu_{k\text{-means}}$	+13.04	Accept
3	P	$\mu_{\text{EvoCluster}} > \mu_{\text{SOM}}$	+16.18	Accept
4	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Hierarchical}}$	+13.67	Accept
5	P	$\mu_{\text{EvoCluster}} > \mu_{\text{EA-based+FS}}$	+10.72	Accept
6	P	$\mu_{\text{EvoCluster}} > \mu_{k\text{-means+FS}}$	+9.88	Accept
7	P	$\mu_{\text{EvoCluster}} > \mu_{\text{SOM+FS}}$	+13.37	Accept
8	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Hierarchical+FS}}$	+11.94	Accept
9	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered EA-based}}$	+8.61	Accept
10	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Reclustered } k\text{-means}}$	+9.28	Accept
11	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered SOM}}$	+11.04	Accept
12	P	$\mu_{\text{EvoCluster}} > \mu_{\text{Re-clustered Hierarchical}}$	+10.36	Accept
Test #	Measure	Null Hypothesis (H_0: $\mu_1 < \mu_2$)	*z*-test	Accept/Reject
1	D	$\mu_{\text{EvoCluster}} < \mu_{\text{EA-based}}$	-7.69	Accept
2	D	$\mu_{\text{EvoCluster}} < \mu_{k\text{-means}}$	-8.25	Accept
3	D	$\mu_{\text{EvoCluster}} < \mu_{\text{SOM}}$	-12.72	Accept
4	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Hierarchical}}$	-8.77	Accept
5	D	$\mu_{\text{EvoCluster}} < \mu_{\text{EA-based+FS}}$	-6.28	Accept
6	D	$\mu_{\text{EvoCluster}} < \mu_{k\text{-means+FS}}$	-6.87	Accept
7	D	$\mu_{\text{EvoCluster}} < \mu_{\text{SOM+FS}}$	-9.93	Accept
8	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Hierarchical+FS}}$	-7.11	Accept
9	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Re-clustered EA-based}}$	-5.98	Accept
10	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Reclustered } k\text{-means}}$	-6.16	Accept
11	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Re-clustered SOM}}$	-8.45	Accept
12	D	$\mu_{\text{EvoCluster}} < \mu_{\text{Re-clustered Hierarchical}}$	-6.97	Accept

Biological Interpretation:

Based on the clustering results obtained by EvoCluster, we are able to discover some interesting associations that may have great biological significance. For example, for *Dataset* 1, when k=5 (which gives the best results), and for *Dataset* 2, when k=6 (which gives the best results), we discovered some associations (represented in if-then rule format) as shown in Tables 5.15 and 5.16, respectively.

TABLE 5.15
INTERESTING ASSOCIATIONS DISCOVERED
(DATASET 1)

If Cond4 = [-2.56, -1.28] then C1 [0.96]	If Cond11 = [-1.53, -0.27] then C4 [0.90]
If Cond8 = [-0.28, 1.01] then C5 [0.90]	If Cond10 = [1.48, 2.97] then C1 [0.92]
If Cond12 = [0.87, 1.95] then C2 [0.90]	If Cond17 = [-0.23, 0.86] then C3 [0.86]
If Cond11 = [0.99, 2.26] then C2 [0.86]	If Cond1 = [-0.14, 1.25] then C5 [0.88]
If Cond3 = [-2.87, -1.41] then C3 [0.94]	If Cond3 = [0.02, 1.45] then C4 [0.92]

TABLE 5.16
INTERESTING ASSOCIATIONS DISCOVERED
(DATASET 2)

If Cond5 = [0.45, 2.12] then C1 [0.86]	If Cond45 = [0.56, 3.35] then C5 [0.85]
If Cond63 = [0.32, 1.32] then C6 [0.90]	If Cond29 = [-2.5, -0.42] then C4 [0.92]
If Cond67 = [-1.27, -0.29] then C3 [0.92]	If Cond10 = [-2.24, -0.21] then C4 [0.88]
If Cond58 = [-1.83, -0.32] then C5 [0.90]	If Cond8 = [0.04, 0.24] then C2 [0.86]
If Cond15 = [-2.17, -0.33] then C6 [0.84]	If Cond51 = [0.56, 3.35] then C1 [0.88]
If Cond36 = [0.43, 1.96] then C2 [0.83]	If Cond21 = [0.53, 2.54] then C3 [0.84]

The discovered associations can be interpreted as follows. In Table 5.15, the rule "If Cond3 = [-2.87, -1.41] then C3 [0.94]", means that if the expression value of a gene under experimental condition, Cond3, is within the interval from [-2.87, -1.41], then there is a probability of 0.94 that it belongs to cluster C3. In Table 5.16, for the rule "If Cond5 = [0.45, 2.12] then C1 [0.86]", it states that if the expression value of a gene under experimental condition, Cond5, is within the interval [0.45, 2.12], then there is a probability of 0.86 that it belongs to cluster C1.

The discovery of these associations is of biological significance in several ways:

1. Based on them, we found that genes within a cluster share similar expression patterns. For example, for *Dataset 2*, genes in cluster C2 expressed very

similarly to each other under the conditions of Cond8 and Cond36, and genes in cluster C3 expressed very similarly to each other under the conditions of Cond21 and Cond67, etc.

2. The associations discovered in each cluster can lead to the discovery of functionally similar or related genes. For example, by closely examining the results with *Dataset 2*, we found that genes such as YBL023C, YEL032W, YLR103C, YLR274W, YBR202W, and YPR019W, etc., which are directly involved in DNA replication [19], [42], satisfied the rule "If Cond10 = [-2.24, -0.21] then C4 [0.88]". We also found that many genes that are involved in mitosis, such as YPR119W, YGR108W, YDR146C, YGR109C, and YML027W [19] satisfied the rules "If Cond63 = [0.32, 1.32] then C6 [0.90]".
3. Biologists can make use of these associations to classify other newly found genes of the same organism in order to infer their potential biological functions [109].
4. In addition to the possible identification of functionally related genes, the discovered associations are expected to help biologists better understanding their expression data. For example, they can help biologists better planning and designing their experiments by focusing on the transcriptional responses of genes in one cluster at a time and by reducing the number of experimental tests required [144].
5. Given that the associations discovered in each cluster are different, we attempted to see if there are any known binding sites in each discovered cluster. To do so, we looked at the corresponding promoter regions (from SGD [19]) of the genes in each cluster. We used a popular motif-discovery

algorithm described in [78] to try to search for transcription factor binding sites in the DNA sequences. Since many regulatory sites can be detected with hexanucleotide analysis [78], we also set the oligonucleotide length to be six. All discovered sites in each cluster were then checked against the well-known binding sites [19], [79]. As shown in Tables 5.17 and 5.18, we did discover the patterns that are known transcription factors binding sites. Moreover, in addition to known binding sites, we were able to discover some other potentially important sites (Tables 5.19 and 5.20). The validity of these sites can be confirmed by biologists using different bio-chemical methods [145].

TABLE 5.17
KNOWN TRANSCRIPTION FACTOR BINDING SITES
REVEALED FROM THE DISCOVERED CLUSTERS
(DATASET 1)

Cluster	*Sequence revealed*	*Binding Site Name*
C1	TAAACA	Mcm1
C2	CTGTCC (potential variant of CTGTGG)	Met31;Met32
C3	CCAGCA	Swi5;Ace2
C4	AAGAAA	SCB
C5	ACGCGT	MCB

TABLE 5.18
KNOWN TRANSCRIPTION FACTOR BINDING SITES
REVEALED FROM THE DISCOVERED CLUSTERS
(DATASET 2)

Cluster	*Sequence revealed*	*Binding Site Name*
C1	ACGCGT	MCB
C2	CGCGAA	SCB
C3	CCAGCA	Swi5;Ace2
C4	CCCAAA	Mcm1
C5	CTGTGG	Met31;Met32
C6	AAACAA	SFF

TABLE 5.19
SOME POTENTIAL TRANSCRIPTION FACTOR
BINDING SITES REVEALED FROM THE DISCOVERED CLUSTERS
(DATASET 1)

Cluster	*Sequence revealed*	*Cluster*	*Sequence revealed*
C1	GATGCC	C4	AGGAAA
	CTCGAC		AGACCA
	AGAAAC		CTCTAA
C2	TGGACA	C5	GTCGCG
	GGTGAT		CGCGTT
	TGTCCA		CGACGC
C3	CCAGCC		
	AGATCG		
	AGGTGA		

TABLE 5.20
SOME POTENTIAL TRANSCRIPTION FACTOR
BINDING SITES REVEALED FROM THE DISCOVERED CLUSTERS
(DATASET 2)

Cluster	*Sequence revealed*	*Cluster*	*Sequence revealed*
C1	AACTCG	C4	GGTCAA
	ACGCGA		TTGGGT
	GCGTTT		TAGGAA
C2	GACGCG	C5	GGCCCA
	TCATGG		TGGATG
	ATCGTC		TCCAAG
C3	GAGCCA	C6	GCTAGA
	TGGTTT		CCACAG
	GGCTGG		GTGTGC

Compared with EvoCluster, other clustering algorithms are only able to discover some of the known binding sites in some of the clusters they discovered (Tables 5.21 and 5.22). This is an indication that the cluster groupings discovered by EvoCluster are more biologically meaningful and significant than the groupings discovered by others. The total numbers of confirmed and suspected binding sites discovered in the clusters found by the various clustering algorithms are also given in Tables 5.23 and 5.24 for *Datasets 1* and *2* respectively. In both data sets, EvoCluster is able to find many more such binding sites.

TABLE 5.21
DISCOVERY OF KNOWN TRANSCRIPTION FACTOR BINDING SITES
IN EACH CLUSTER DISCOVERED BY DIFFERENT CLUSTERING ALGORITHMS
(DATASET 1) (SITE NAME AND THE NO. OF OCCURRENCES)

	C1	*C2*	*C3*	*C4*	*C5*
EvoCluster	Mcm1 (64)	Met31;Met32 (99)	Swi5;Ace2 (32)	SCB (138)	MCB (101)
EA-based	Mcm1 (30)	Met31;Met32 (58)	-	SCB (102)	MCB (73)
k-means	Mcm1 (23)	-	Swi5;Ace2 (21)	SCB (119)	MCB (85)
SOM	-	-	Swi5;Ace2 (25)	SCB (114)	MCB (65)
Hierarchical	Mcm1 (26)	-	-	SCB (96)	MCB (57)

TABLE 5.22
DISCOVERY OF KNOWN TRANSCRIPTION FACTOR BINDING SITES
IN EACH CLUSTER DISCOVERED BY DIFFERENT CLUSTERING ALGORITHMS
(DATASET 2)

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
EvoCluster	MCB (139)	SCB (86)	Swi5;Ace2 (44)	Mcm1 (62)	Met31;Met32 (102)	SFF (207)
EA-based	MCB (95)	-	-	Mcm1 (55)	-	SFF (116)
k-means	MCB (79)	-	-	Mcm1 (40)	-	SFF (134)
SOM	-	-	-	Mcm1 (36)	-	SFF (89)
Hierarchical	MCB (61)	SCB (31)	-	-	-	SFF (107)

TABLE 5.23
TOTAL NUMBER OF KNOWN AND POTENTIAL TRANSCRIPTION
FACTOR BINDING SITES IN EACH CLUSTER DISCOVERED
BY DIFFERENT CLUSTERING ALGORITHMS
(DATASET 1)

	C1	*C2*	*C3*	*C4*	*C5*
EvoCluster	14	11	8	17	19
EA-based	9	5	3	7	15
k-means	6	2	5	12	9
SOM	4	3	3	9	13
Hierarchical	4	-	2	3	10

TABLE 5.24
TOTAL NUMBER OF KNOWN AND POTENTIAL TRANSCRIPTION FACTOR BINDING SITES IN EACH CLUSTER DISCOVERED BY DIFFERENT CLUSTERING ALGORITHMS (DATASET 2)

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
EvoCluster	12	21	9	4	24	15
EA-based	5	9	6	4	8	11
k-means	8	7	3	3	13	10
SOM	4	5	-	3	7	6
Hierarchical	5	12	-	-	4	6

5.4 Extension of EvoCluster: Mining Overlapping Clusters in Gene Expression Data

While many clustering algorithms have been used successfully with gene expression data, it should be noted that they usually perform their tasks under the assumption that each gene belongs only to one cluster. Such an assumption can sometimes be an over-simplification of the great biological complexity underlying the gene expression process. As many proteins have multiple functional roles in a cell, they have to interact with different groups of other proteins to fulfill them. The genes that produce these proteins are therefore expected to co-express with different groups of other genes in order to meet the varying demands of a cell. In other words, depending on the experimental conditions being investigated, each gene may have similar expression patterns with different groups of other genes in other clusters and they can, therefore, belong to more than one cluster. This poses a challenge to existing clustering algorithms as they need to tackle two difficult problems: (i) they need to handle overlapping clusters which they were not originally developed to do so, and (ii) they need to discover overlapping clusters in the presence of noise.

In order to do so, some attempts have been made to use the fuzzy *k*-means algorithm [20] in the clustering of gene expression data [9], [58], [63]. The main

difference between the fuzzy *k*-means algorithm and the standard *k*-means algorithm is in the assignment of genes to a cluster. Rather than assigning a gene to one and only one cluster, the fuzzy *k*-means algorithm allows each gene to be assigned partially to more than one cluster according to a degree of membership that ranges between 0 and 1. Genes that are very near a given cluster centroid are assigned a higher degree of membership to that cluster and genes that are very far away are assigned a lower degree. The only constraint such a cluster assignment process needs to work under is that a gene cannot belong completely (with full degree of membership) to more than one cluster. In fact, the sum of the degree of membership for a gene to belong to different clusters has to be 1.

When dealing with gene expression data, it should be noted that, similar to many existing clustering algorithms, the fuzzy *k*-means algorithm may not be able to perform effectively. For example, it makes use of the Euclidean distance or some correlation coefficients when measuring similarity. One problem with these similarity measures is that they do not differentiate between the relevancy of different data values collected under different experimental conditions. And for this reason, they do not give very accurate measurements when dealing with noisy data. Other than these problems, the fuzzy *k*-means algorithm also requires users to define, in advance, a fuzziness parameter, $m \in \{1, \infty\}$, ahead of time as it may require many trials and errors. The fuzziness parameter determines the degree of fuzziness of the clustering process. When *m* is set to 1, the clustering algorithm performs a hard partition and when *m* is set to infinity, the clustering assumes the highest degree of fuzziness. If *m* is not properly set, there is a chance that none of the genes can be tightly associated with any clusters. In such case, even though a clustering structure

can be found, the degree-of-membership values can all be relatively small making the results very difficult to interpret. The selection of appropriate fuzziness parameter is hence very important for the fuzzy *k*-means algorithm to perform effectively but it also adds an additional level of difficulty to the cluster-discovery process. In addition to the above, the fuzzy *k*-means algorithm does not make explicit the patterns discovered in a data set during the clustering process. To better understand and interpret the clustering results, a separate technique is usually required for patterns underlying each discovered cluster to be uncovered explicitly.

In an attempt to solve these problems, we combined EvoCluster [111] (as discussed in Section 5.1) with the re-clustering process [112] (as described in Chapter 4, Section 4.1, phase 2) to mining overlapping clusters in gene expression data. This extended version [116]-[117] also consists of two phases: an initial clustering phase and a second re-clustering phase, and is able to discover overlapping clusters on one hand and overcome some of the limitations of existing methods on the other.

For the initial clustering phase, EvoCluster is used to discover the initial clusters from gene expression data. For the re-clustering phase, interesting associations between the expression levels and cluster labels are first identified in the initial clusters. Then, based on the discovered associations and the weight of evidence measure, rather than assigning a gene to the cluster with the largest total weight of evidence as we performed before, the total weight of evidence supporting a gene belongs to each cluster, C_p, where $p = 1,..., P$ and P is the total number of initial clusters discovered, is calculated (using Eq.(4.3)). The cluster memberships of the genes that have previously been assigned are then re-evaluated to determine whether

they should belong to the same cluster or be assigned to more than one. This extended version of EvoCluster facilitates the discovery of overlapping clusters by assigning a gene to more than one cluster only if there is a positive total weight of evidence of this gene to belong to the given cluster. Moreover, it can also facilitate the identification of groups of genes that have a strong association (i.e., with large weight of evidence) to the cluster for further biological analysis, for example, functional annotations [103].

5.4.1 Experiments

A. Experimental Data

For experimentation, we used a set of simulated data consisting of 300 records each characterized by 50 different attributes that takes on values from [0.0, 1.0]. These records were first grouped into three clusters based on embedding the patterns unique to each cluster. To do so, for each cluster, we randomly select 20% of the attributes. For each selected attribute, its values in 40% of the records in this cluster were generated randomly from within the range $[L_c, U_c]$, where L_c and U_c were also generated randomly so that $0.0 \leq L_c \leq U_c \leq 1.0$. To ensure that overlapping of clusters, three sets of overlapping patterns were embedded into the data as follows. First, for each set of overlapping patterns, 10% of the attributes and 20% of the records were randomly selected (selected from the whole data set rather than from each cluster). The value of a selected attribute in each selected record was then generated randomly only from within the same range $[L_f, U_f]$, where L_f and U_f were also randomly generated and $0.0 \leq L_f \leq U_f \leq 1.0$.

In addition to simulated data, we have also tested the proposed algorithm using two sets of gene expression data. The first set, *Dataset 1*, contains 517 genes whose expression levels vary in response to serum concentration in human fibroblasts (under 12 different experimental conditions). We tried in our experiments to partition this dataset into different clusters from 4 to 8 [83]. The second data set, *Dataset 2*, contains 384 genes whose expression levels were measured under 17 different experimental conditions and we also tried to partition this dataset into different clusters from 4 to 8 [161].

B. Evaluating Criteria

The performance of the proposed algorithm was evaluated using two objective measures: (i) the F-measure and (ii) the silhouette measure.

The F-measure, as discussed in Chapter 4, Section 4.2, combines the idea of "precision" and "recall" from the field of information retrieval. When the correct clustering arrangement of a set of data is known, the F-measure can be used to determine how well a discovered clustering arrangement compares with that of the correct, original one. The F-measure, it should be noted, can be used with clusters having fuzzy boundaries. To do so, the clusters are first converted into crisp boundaries by assigning records to the clusters they belong to with the largest degree of memberships [96].

The silhouette measure [135] calculates the silhouette value of a gene, g_i, which reflects the likelihood of g_i belonging to a cluster C_p. It does so by first estimating two scalars $a(g_i)$ and $b(g_i)$ where $a(g_i)$ is the average distance between g_i and all other genes in C_p and $b(g_i)$ is the smallest of $d(g_i, C_q)$, where $d(g_i, C_q)$ is

defined to be the average distance of g_i to all genes in C_q, $C_p \neq C_q$. The silhouette $s(g_i)$ of g_i is then defined to be the ratio, $\frac{b(g_i) - a(g_i)}{\max\{a(g_i), b(g_i)\}}$. The silhouette value lies between –1 to 1. When its value is less than zero, the corresponding gene is poorly classified. The overall silhouette value of a cluster is the average of $s(g_i)$ of all the genes in the cluster [123].

5.4.2 Results and Discussions

For comparison, we compared the performance of the proposed algorithm with the fuzzy *k*-means [20] algorithm using the simulated and real expression data as described above. In our experiments with the fuzzy *k*-means algorithm, to ensure that its performance is not affected by poor choice of initial cluster centroid, we performed 100 runs with each using different randomly-generated initial cluster centroids and also, such 100-run test was repeated 10 times. The 10 best results obtained from each 100-run test were then recorded. In addition, the fuzziness parameter, *m*, used in our experiment was set to different values ranging from 1.1 to 2 as suggested in [20] and the *m* that gave us the best clustering result was selected. For the simulated data, *m* happens to give us the best result in terms of the F-measure when set to 1.1. In order to improve the performance of the fuzzy *k*-means algorithm, we also performed feature selection for it. This feature selection procedure is same as those used by other clustering algorithms as discussed in Section 5.3. But, one additional step performed for the fuzzy *k*-means algorithm is that the fuzzy clusters were first converted into crisp boundaries by assigning records to the clusters they belong to with the largest degree of memberships [99].

For EvoCluster, the parameter settings, including population size, number of reproductions, etc., are showed in Table 5.25. With these parameter settings, the result we obtained using the simulated data for testing is showed in Table 5.26. It should be noted that the number of clusters (k=3) to be discovered was known-in-advance for the simulated data, the length of the *chromosome* was fixed in our experiment to be 3

TABLE 5.25
PARAMETER SETTINGS OF EVOCLUSTER USED IN SIMULATED DATA
(P_c REPRESENTS THE PROBABILITY OF GUIDED OR UNGUIDED CROSSOVER OPERATOR SELECTED AND P_m REPRESENTS THE PROBABILITY OF GUIDED OR UNGUIDED MUTATION OPERATOR SELECTED)

	Pop. Size	*MIN*	*MAX*	P_c	P_m	L_g	U_g	L_r	U_r
Simulated data	50	3	3	0.5	0.5	0.2	0.8	0.2	0.8

TABLE 5.26
COMPARISON OF THE AVERAGE F-MEASURE
(SIMULATED DATA)

	Proposed	*Fuzzy k-means*	*Fuzzy k-means + FS*
Average	0.82	0.60	0.71

As shown in the above table, compared with both fuzzy *k*-means algorithms, the proposed algorithm performs better. Also, it seems that the performances of the fuzzy *k*-means algorithms are not good when handling very noisy data. In addition, the associations discovered by the proposed algorithm can reveal the overlapping patterns embedded in the clusters. For example, the associations (represented in if-then rule format) “If Cond15 = [0.07, 0.25] then C1” and “If Cond15 = [0.07, 0.25] then C2” with the same conditional part were discovered in clusters C1 and C2 respectively, this indicates that these clusters contain the overlapping pattern [117].

For gene expression data, the parameter settings of EvoCluster are showed in Table 5.27.

TABLE 5.27
PARAMETER SETTINGS OF EVOCLUSTER USED
IN GENE EXPRESSION DATA

	Pop. Size	*MIN*	*MAX*	P_c	P_m	L_g	U_g	L_r	U_r
Dataset 1	50	4	8	0.5	0.17	0.2	0.8	0.2	0.8
Dataset 2	50	4	8	0.5	0.17	0.2	0.8	0.2	0.8

For *Dataset 1*, 4 clusters (which gives the best result), $C_1^{(1)}$, $C_2^{(1)}$, $C_3^{(1)}$, $C_4^{(1)}$ of sizes 186, 116, 141, 74, respectively, were discovered after Phase 1. For *Dataset 2*, 5 clusters (which gives the best result), $C_1^{(2)}$, $C_2^{(2)}$, $C_3^{(2)}$, $C_4^{(2)}$, $C_5^{(2)}$ of sizes 65, 106, 83, 74, 56, respectively, were discovered after Phase 1. The clustering results obtained for *Datasets 1* and *2* after Phase 1 are visualized in Figs. 5.2 and 5.3 respectively. From these figures, it is noticed that EvoCluster (used in Phase 1) is effective in identifying the initial clusters.

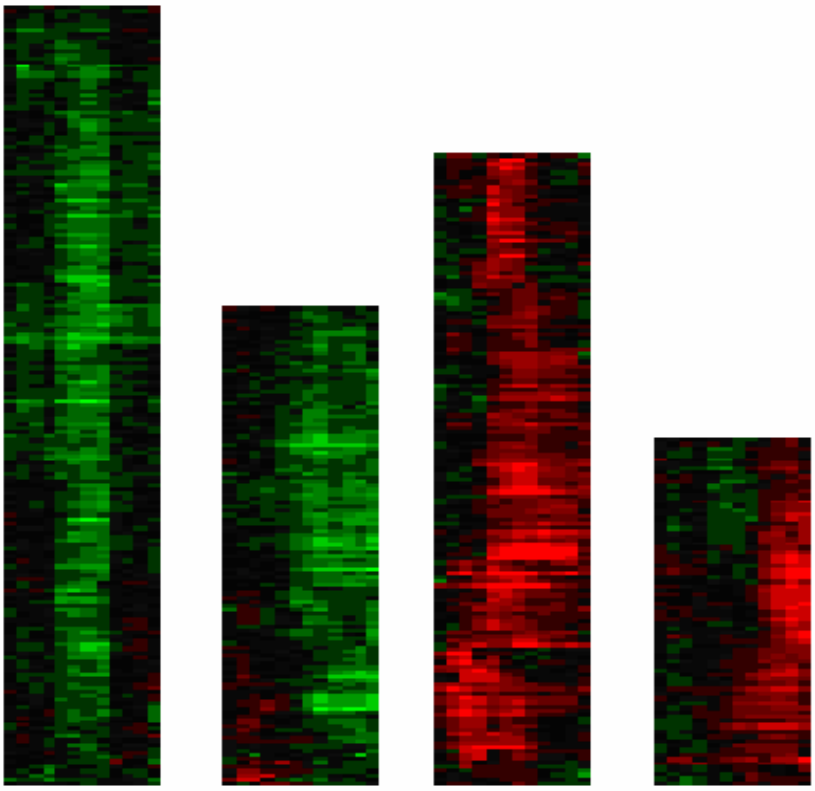

Fig. 5.2 The four initial clusters discovered - Dataset 1. (From left to right: $C_1^{(1)}$ has 186 genes, $C_2^{(1)}$ has 116 genes, $C_3^{(1)}$ has 141 genes and $C_4^{(1)}$ has 74 genes.)

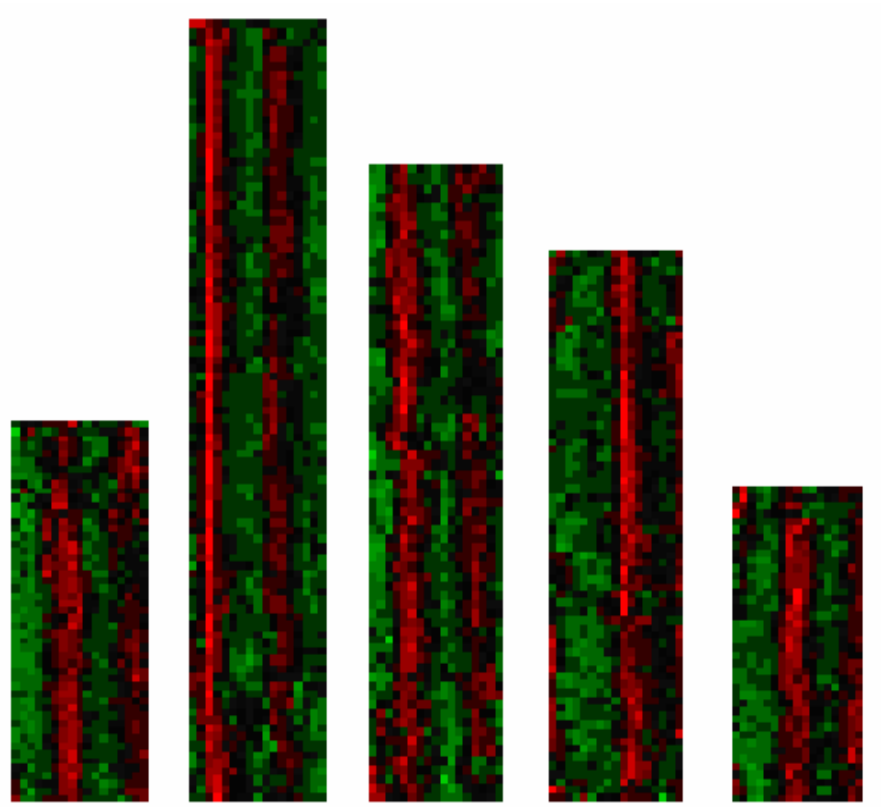

Fig. 5.3 The five initial clusters discovered - *Dataset* 2. (From left to right: $C_1^{(2)}$ has 65 genes, $C_2^{(2)}$ has 106 genes, $C_3^{(2)}$ has 83 genes, $C_4^{(2)}$ has 74 genes, and $C_5^{(2)}$ has 56 genes.)

Based on the initial clusters discovered in Phase 1, the second re-clustering phase was performed. During the re-clustering process, interesting associations were discovered in each initial cluster. Based on these findings, the cluster membership of each gene in each cluster was re-evaluated to determine if it should remain in the same cluster or be assigned to more than one cluster. Figs. 5.4 and 5.5 below show the clustering results after the re-clustering process. The overlapping region between different clusters is marked in the figure as $c_{jk}^{(i)}$ which represents the overlapping subset between the clusters $C_j^{(i)}$ and $C_k^{(i)}$. Similarly, the non-overlapping region of each cluster is marked in the figure as $c_j^{(i)}$ which represents the non-overlapping subset of the cluster $C_j^{(i)}$. As each gene is allowed to belong to more than one cluster,

the sizes of the overlapping clusters are larger than their corresponding initial clusters.

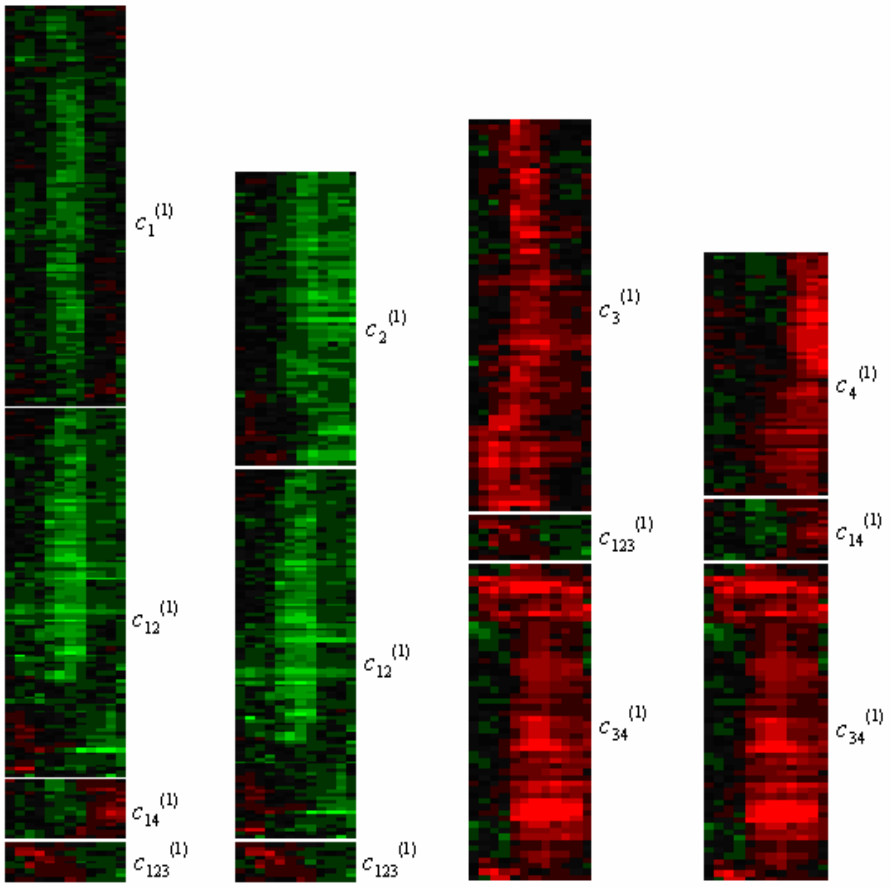

Fig. 5.4 The four overlapping clusters discovered by the proposed algorithm (*Dataset 1*). (From left to right: $C_1^{(1)}$ has 219 genes, $C_2^{(1)}$ has 159 genes, $C_3^{(1)}$ has 167 genes and $C_4^{(1)}$ has 128 genes. The label beside the image of each cluster indicates the overlapping and non-overlapping regions, i.e., $c_1^{(1)}$ labels the non-overlapping region in $C_1^{(1)}$, and $c_{12}^{(1)}$ labels the overlapping region between $C_1^{(1)}$ and $C_2^{(1)}$.)

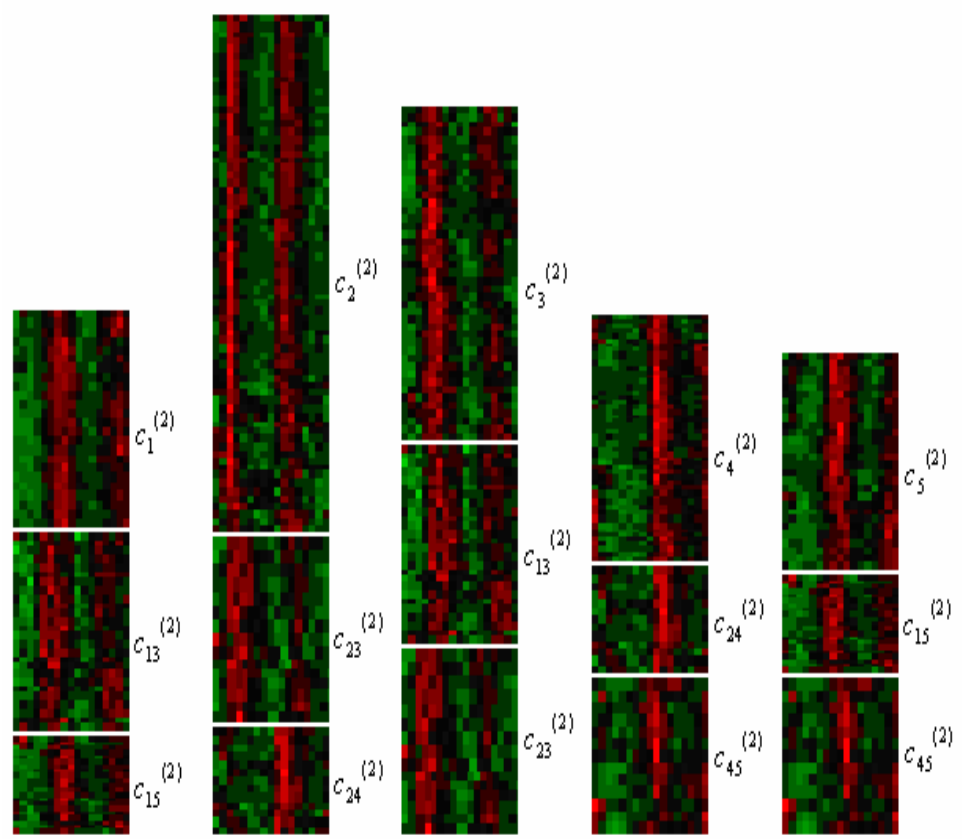

Fig. 5.5 The five overlapping clusters discovered by the proposed algorithm (*Dataset 2*). (From left to right: $C_1^{(2)}$ has 91 genes, $C_2^{(2)}$ has 125 genes, $C_3^{(2)}$ has 104 genes, $C_4^{(2)}$ has 86 genes, and $C_5^{(2)}$ has 77 genes.)

From Figs. 5.4 and 5.5, one can see that the genes discovered in each overlapping region have expression patterns similar to that with other genes in the non-overlapping regions. The results show that the re-clustering process is effective in discovering overlapping clusters from gene expression data.

In addition to the above, we have also compared the clusters discovered by the proposed algorithm with those discovered by the fuzzy *k*-means algorithms using the silhouette measure. In our experiments, the genes belonging to the same cluster were treated as positive training instances whereas genes that do not belong to the cluster at all were treated as negative training instances. And also, for the fuzzy *k*-means algorithms, the fuzziness parameter (*m*) was set to 1.3 for *Dataset* 1 and 1.2 for

Dataset 2 as such parameter settings allow them to perform at its best in terms of the silhouette measure.

According to the Tables 5.28 and 5.29, we found that the qualities of the clusters discovered by the proposed algorithm have better silhouette values than those discovered by the fuzzy *k*-means algorithms in both data sets.

TABLE 5.28
COMPARISON OF THE AVERAGE SILHOUETTE MEASURE BETWEEN THE PROPOSED ALGORITHM AND THE FUZZY K-MEANS ALGORITHMS (DATASET 1)

	Proposed	*Fuzzy k-means*	*Fuzzy k-means + FS*
k=4	0.53	0.40	0.46
k=5	0.50	0.36	0.43
k=6	0.48	0.33	0.39
k=7	0.40	0.30	0.32
k=8	0.43	0.28	0.34
Avg	0.47	0.33	0.39

TABLE 5.29
COMPARISON OF THE AVERAGE SILHOUETTE MEASURE BETWEEN THE PROPOSED ALGORITHM AND THE FUZZY K-MEANS ALGORITHMS (DATASET 2)

	Proposed	*Fuzzy k-means*	*Fuzzy k-means + FS*
k=4	0.44	0.33	0.36
k=5	0.49	0.37	0.41
k=6	0.46	0.31	0.36
k=7	0.38	0.21	0.29
k=8	0.35	0.23	0.26
Avg	0.42	0.29	0.34

For further performance evaluation, the gene expression data sets were corrupted by adding uniformly generated random noise to every gene expression profile [51]. Figs. 5.6 and 5.7 show how the proposed algorithm compares with the fuzzy *k*-means algorithms on the corrupted gene expression data. Based on the discovered results, we found that the proposed algorithm, in spite of the additionally added noise at various levels, still outperforms the fuzzy *k*-means algorithms.

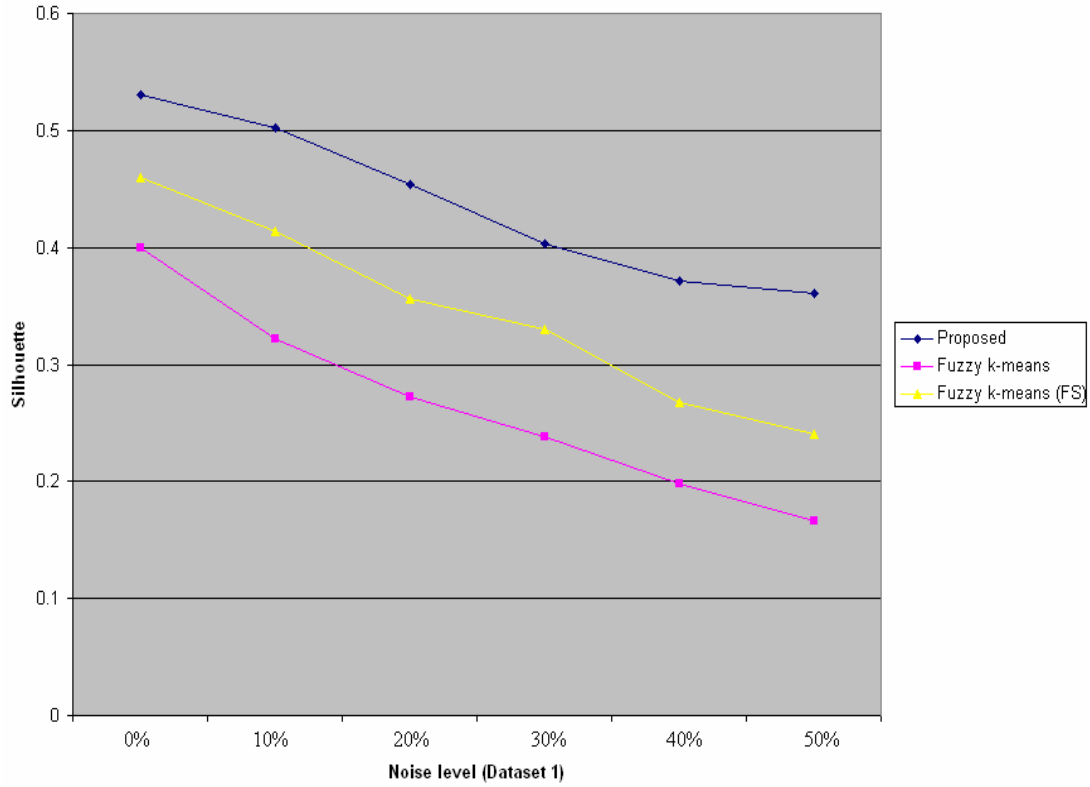

Fig. 5.6 Comparison of the clustering performance (silhouette measure) in a noisy environment (*Dataset 1*, *k*=4). A noise level at 10% means that 10% of each gene expression profile consists of added noise.

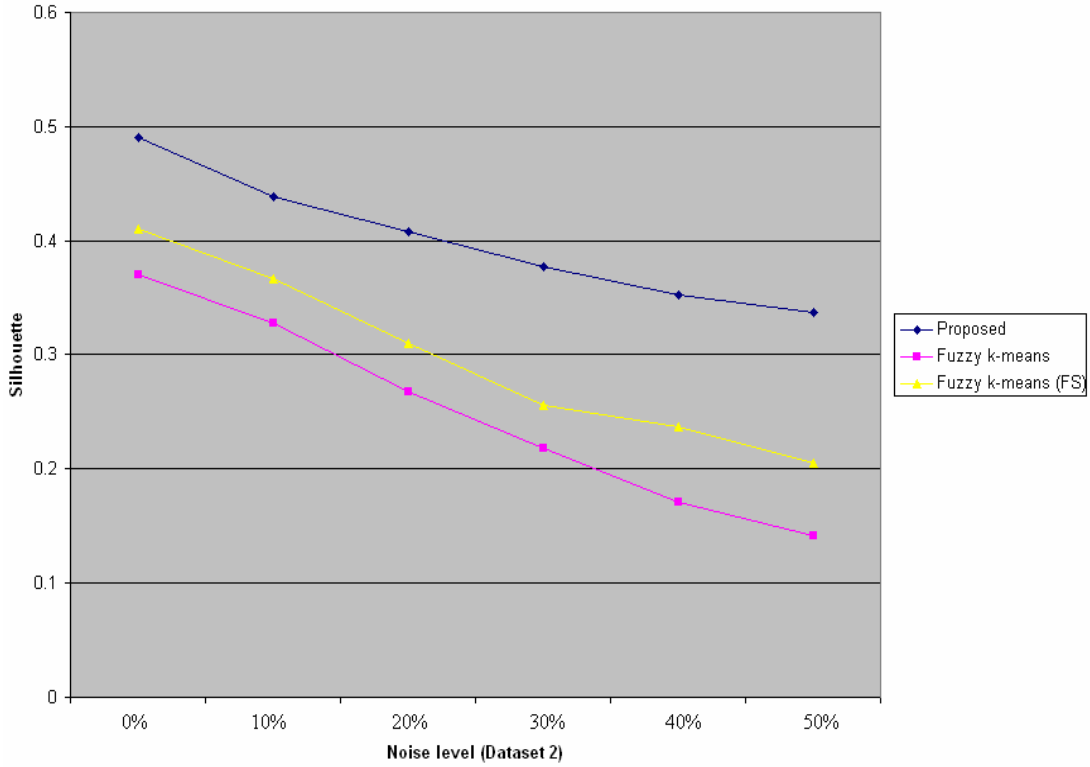

Fig. 5.7 Comparison of the clustering performance (silhouette measure) in a noisy environment (*Dataset 2*, *k*=5).

Biological Interpretation:

Other than evaluating the results statistically, we have also evaluated the clustering results according to their biological functions. Since genes that have similar expression patterns may have similar or related biological functions [38] and it is shown in [148] that significant enrichment of genes belonging to the given functional categories can be revealed in the clusters discovered through clustering. Therefore, we also evaluated the results according to the biological functions of genes that can be discovered in each cluster. To evaluate the effectiveness of the proposed algorithm, we therefore look at the percentage of genes in each function category discovered in the initial non-overlapping clusters after Phase 1 to see if there is a corresponding increase in the overlapping clusters discovered after Phase 2.

When comparing the clusters discovered in the data (*Dataset 2*) after Phase 1 with those discovered after Phase 2 based on the MIPS functional catalogue database [104], we found that in each overlapping cluster, the percentage of genes in each functional category is greater than that obtained in the corresponding initial cluster (Table 5.30). Also, the *p-value* associated with each functional category discovered in the overlapping cluster is smaller than that obtained in the corresponding initial cluster (the *p-value* is calculated to obtain the chance probability of observing a set of genes from a particular MIPS functional category within a cluster, thus low *p-value* indicates high significance [104]). This indicates that the discovered overlapping clusters are biologically significant.

TABLE 5.30
COMPARISON OF THE ENRICHMENT OF GENES IN EACH FUNCTIONAL CATEGORY BETWEEN THE INITIAL (PHASE 1) AND OVERLAPPING (PHASE 2) CLUSTERS (DATASET 2)

	MIPS Functional Category	*Phase 1 (%)*	*Phase 1 (p-value)*	*Phase 2 (%)*	*Phase 2 (p-value)*
$C_1^{(2)}$	BUD/GROWTH TIP	7.83%	0.42	13.72%	0.29
	MITOCHONDRION	18.14%	0.21	33.46%	0.04
	ENDOPLASMIC RETICULUM	10.22%	0.07	14.30%	0.03
$C_2^{(2)}$	TRANSPORTED COMPOUNDS	8.29%	0.35	10.46%	0.21
	DNA PROCESSING	11.73%	0.38	14.30%	0.23
$C_3^{(2)}$	EUKARYOTIC PLASMA MEMBRANE/ MEMBRANE ATTACHED	5.95%	0.41	7.88%	0.20
	FUNGAL/ MICROORGANISMIC CELL TYPE DIFFERENTIATION	5.67%	0.21	10.28%	0.04
	C-COMPOUND AND CARBOHYDRATE METABOLISM	6.16%	0.23	9.92%	0.14
$C_4^{(2)}$	CELL GROWTH/ MORPHOGENESIS	7.08%	0.40	8.57%	0.29
	CELLULAR SENSING AND RESPONSE	5.32%	0.22	7.66%	0.09
	TRANSPORT ROUTES	4.96%	0.31	6.18%	0.28
$C_5^{(2)}$	RNA SYNTHESIS	12.73%	0.29	16.87%	0.11
	NUCLEUS	19.46%	0.22	24.18%	0.18
	CYTOSKELETON	15.81%	0.59	24.60%	0.36

5.5 Summary Remarks

With the advent of microarray technology, we are now able to monitor simultaneously the expression levels of thousands of genes during important biological processes. Due to the large number of data collected everyday and due to the very noisy nature in the data collection process, interpreting and comprehending the experimental results has become a big challenge. In this chapter, we have proposed a novel evolutionary clustering algorithm called EvoCluster. EvoCluster encodes an entire cluster grouping in a *chromosome* so that each *gene* encodes one cluster. Based on such a structure, it makes use of a set of reproduction operators to facilitate the exchange of grouping information between *chromosomes*. The fitness function it adopts is able to differentiate between how relevant the expression level is in determining a particular cluster grouping. As such, instead of just local pair-wise distances, it also takes into consideration how clusters are arranged globally. Moreover, it does not require the number of clusters to be decided in advance, and the associations discovered in each cluster can be explicitly revealed and presented for easy interpretation. In addition, we have also proposed the possible extension of EvoCluster to mining overlapping clusters in gene expression data.

Experimental results using both simulated and real data show that the proposed algorithms are very robust in the presence of noise. They are able to search for near optimal solutions effectively, and discover interesting associations in the noisy data for meaningful groupings. The results also show that, under some common performance measures, the proposed algorithms are better than other algorithms commonly used in gene expression data analysis, and the discovered clusters contain more biologically meaningful patterns. In particular, we could correlate the clusters

of co-expressed genes discovered to their DNA sequences, and found that we were able to uncover known and new biological binding sites in each cluster of co-expressed genes.

Compared with other clustering algorithms such as *k*-means, SOM or fuzzy *k*-means, EvoCluster is about 12-18 times slower than others when the time it takes for a reproduction to be performed is compared against the time it takes for performing an iteration in each of these algorithms. However, as shown in the experiments above, even if more computational resources (in terms of the number of iterations) are given to other clustering algorithms, EvoCluster will likely be giving the best clustering results. For microarray analysis, since the results are normally not required immediately, the relatively longer evolutionary process that EvoCluster takes to find a better solution is not very important. In order to cope with very large gene expression data sets, the inherently parallel nature of the problem solving process of EvoCluster can be exploited.

Chapter 6

Conclusions

6.1 Summary

Gene expression data mining as a new research area poses new challenges to data mining researchers. Gene expression data are typically very noisy and have very high dimensionality. To tackle bioinformatics problems involving them, traditional data mining techniques may not be the best tools to use as they were not originally developed to deal with such data. For this reason, the contributions of this thesis are to propose some data mining techniques to solve these problems effectively. In particular, these techniques can be used to solve the problems of reconstructing gene regulatory networks (GRNs) and clustering gene expression data. The former is concerned with the problem of discovering gene interactions to infer the structures of gene regulatory networks. The latter is concerned with the problem of discovering clusters of co-expressed genes so that genes that have similar expression patterns under different experimental conditions can be identified.

To reconstruct GRNs, we have proposed to use an association-discovery technique [113]-[115], which is based on residual analysis and an information theoretic measure, for the effective inference of the structures of GRNs from time-dependent gene expression data. This association-discovery technique can discover interesting association relationships between genes in high-dimensional and very noisy expression data without the need for additional feature selection procedures. By computing an average gene expression value which serves as a reference point

for how large the value is, the proposed technique can discover interesting sequential associations between genes such as “if a gene is highly expressed, its dependent gene is then lowly expressed in the next time point”, etc. Based on these findings, the user not only can determine those genes affecting a target/dependent gene and also can identify whether or not the target gene is supposed to be activated or inhibited. In addition, the sequential associations discovered can also be used to predict how a gene would be affected by other genes from the unseen samples. Experimental results on real expression data show that the proposed technique can be very effective and the discovered sequential associations reveal known gene regulatory relationships that could be used to infer the structures of GRNs. One additional advantage of the reconstruction of GRNs using the association-discovery technique is that the user can easily improve the classifier by adding new expression data and reproduce underlying structures of a network consistent with the data. Since such iterative improvements can be part of an interactive process. Therefore, the proposed technique can be considered as a basis for an interactive expert system for GRNs reconstruction.

Given clusters (or classes) of genes, the association-discovery technique proposed can also be used to construct classifiers by finding interesting association relationships between gene expression levels and cluster (or class) labels. Based on discovering such relationships, we have developed a two-phase clustering algorithm [112], [118], [120] for gene expression data. This algorithm consists of an initial clustering phase and a second re-clustering (or re-classification) phase. In the first phase, existing clustering algorithm, such as *k*-means or a hierarchical clustering algorithm, can be applied. The results, which consist of a number of initial clusters, can then be used for re-clustering. The re-clustering problem can be formulated as a

classification problem by treating the data in each initial cluster as training data for the construction of a classifier. Once the classifier is constructed, the genes in each initial cluster can then be re-classified either into the same cluster or into different clusters. It should be noted that the re-clustering phase allows for probabilistic associations to be detected. It performs its task by distinguishing between relevant and irrelevant expression levels and by doing so, it takes into consideration global information contained in a specific cluster arrangement by evaluating the importance of different expression levels in determining cluster memberships. This feature makes the proposed algorithm more robust to noisy data when compared to those existing methods that only rely on local pair-wise similarity measures. In addition, the discovered associations indicate how relevant the expression levels under a particular set of experimental conditions are in a particular cluster and are made explicit for possible interpretation. Experimental results on both simulated and real data show that the proposed two-phase clustering algorithm can be very effective for discovering clusters in the presence of noisy data. It is able to assign genes, whose cluster memberships cannot be easily determined by existing clustering methods, into the appropriate clusters. When identifying regulatory motifs at the promoter regions of the co-expressed genes in the discovered clusters, the known transcription-factor binding sites specific to each cluster can be discovered. These binding sites can provide explanations for the co-expressed patterns.

Since the effectiveness of the two-phase clustering algorithm depends, to some extent, on that of the existing clustering method used in the first phase, we have developed a novel evolutionary clustering algorithm, called EvoCluster [111], [119], [121], that can be used in the first phase to overcome some of the limitations of existing ones. It not only is able to perform well in the presence of very noisy data, it

can also be used to discover overlapping clusters. EvoCluster makes use of an evolutionary approach to guide the search for optimal or near-optimal clustering arrangement. To do so, it encodes the entire cluster grouping in a *chromosome* so that each *gene* encodes one cluster and each cluster contains the labels of the data records grouped into it. Then, given the encoding scheme, it has a set of special crossover and mutation operators that facilitates the exchange of grouping information between two *chromosomes* on one hand and allows variation to be introduced to avoid trapping at local optima on the other. For fitness evaluation, EvoCluster makes use of the association-discovery technique to discover interesting association relationships in each possible cluster to determine how good the cluster arrangement encoded in a *chromosome* is. Unlike many similarity measures that are based on local pair-wise distances that may not give very accurate measurements in the presence of very noisy data, the proposed fitness measure is probabilistic and it takes into consideration global information contained in a particular grouping of data. It is able to distinguish between relevant and irrelevant expression levels in the data during the clustering process, and explain clustering results by explicitly revealing interesting associations discovered in each cluster. In addition, there is no requirement for the number of clusters to be decided in advance. In an attempt to discover overlapping clusters in noisy gene expression data, we have also developed the extended version of EvoCluster. This extended version [116]-[117] consists of two phases: an initial clustering phase and a second re-clustering phase, and is able to discover overlapping clusters on one hand and overcome some of the limitations of existing methods on the other. For the initial clustering phase, EvoCluster is used to discover the initial clusters from gene expression data. For the re-clustering phase, interesting associations between the expression levels and cluster labels are first

identified in the initial clusters. Then, based on the discovered associations, rather than assigning a gene to the cluster with the largest total weight of evidence, the total weight of evidence supporting a gene belongs to each cluster is calculated. The cluster memberships of the genes that have previously been assigned are then re-evaluated to determine whether they should belong to the same cluster or be assigned to more than one. This extended version of EvoCluster facilitates the discovery of overlapping clusters by assigning a gene to more than one cluster only if there is a positive total weight of evidence of this gene to belong to the given cluster. Moreover, it can also facilitate the identification of groups of genes that have a strong association (i.e., with large weight of evidence) to the cluster for further biological analysis, for example, functional annotations. Experimental results on both simulated and real data show that the proposed algorithms are very robust in the presence of noise. They are able to search for near optimal solutions effectively, and discover interesting associations in the noisy data for meaningful groupings. The results also show that, under some common performance measures, they are better than other existing methods commonly used in gene expression data analysis, and the discovered clusters contain more biologically meaningful patterns. In particular, we could correlate the co-expressed genes discovered to their DNA sequences, and found that we were able to uncover known and new biological binding sites in each cluster of co-expressed genes.

6.2 Future Work

To handle continuous values, the proposed association-discovery technique has to perform discretization. Since the crisp discretization procedure it relies on does not take into account the expression values at the interval boundaries. These values may

end up assigned to different intervals even though they are very similar. This may add noise to the data and result in some important patterns being overlooked. For this reason, rather than crisp discretization, fuzzy discretization [101] of gene expression values can be exploited. To do so, the quantitative expression values need to be transformed into linguistic variables and terms by some pre-defined membership functions. Since, the association-discovery technique can be easily modified to handle the degree of membership. Therefore, given a set of time-dependent fuzzy data, the fuzzy association relationships between genes can also be discovered. By applying this fuzzy data mining technique to GRNs reconstruction, we hope that the prediction accuracy can be improved and also more known gene regulatory relationships can be discovered.

To classify proteins into functional families based on their primary sequences, existing classification methods such as the *k*-NN, HMM and SVM-based algorithms are often used [54]. For most of these algorithms to perform their tasks, protein sequences need to be properly aligned first. Since the alignment process is error-prone, protein classification may not be performed very accurately. In addition to the request for accurate alignment, many existing methods require additional techniques to decompose a protein multi-class classification problem into a number of binary problems. This may slow the learning process when the number of classes being handled is large. To increase both efficiency and accuracy, the proposed association-discovery technique can also be applied. Using the sliding window approach, each protein sequence is first converted into a number of subsequences with equal length. Then, the association-discovery technique can be used to mine a set of subsequences without having to go through a sequence-alignment process. And also, the association-discovery technique is able to discover hidden patterns unique to each

protein functional family by making use of residual analysis that can determine whether or not a protein residue is useful for the characterization of a class (family). Given a set of conserved sequences, it is well-known that it is very difficult, on the basis of multiple alignment of protein sequences alone, to determine which residues of a protein are important for its functional or structural characterization. By being able to uncover hidden patterns for possible interpretation, the association-discovery technique makes such task much easier. Based on the discovered patterns unique to each protein functional family, we believe that they can lead to better understanding of protein functions and can also allow functionally significant structural features of different protein families to be better characterized.

Besides gene expression data, EvoCluster can be used to cluster other biological data such as DNA and protein sequence data, and most importantly, it can also be used for solving different kinds of clustering problems in other application areas. Moreover, in order to efficiently cluster large data sets, the inherently parallel nature of the problem solving process of EvoCluster can be exploited. And also, other possible alternatives, such as simulated annealing [127], could be explored. For future work, we intend to look into how specifically these can be done and compare EvoCluster with other EA-based clustering methods that have emerged very recently [69].

References

1. Agrawal, R., Imielinski, T. and Swami, A., “Mining association rules between sets of items in large databases,” *in Proc. of ACM SIGMOD International Conference on Management of Data*, pp. 207-216, 1993.

2. Agrawal, R. and Srikant, R., “Fast algorithms for mining association rules in large databases,” *in Proc. of International Conference on Very Large Databases*, pp. 487-499, 1994.

3. Altschul, S.F., Gish, W., Miller, W., Myers, E.W. and Lipman, D.J., “Basic local alignment search tool,” *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403-410, 1990.

4. Akutsu, T., Miyano, S. and Kuhara, S., “Identification of genetic networks from a small number of gene expression patterns under the boolean network model,” *Pacific Sym. on Biocomputing*, pp. 17-28, 1999.

5. Akutsu, T., Miyano, S. and Kuhara, S., “Inferring qualitative relations in genetic networks and metabolic pathways,” *Bioinformatics*, vol. 16, no. 8, pp. 727-734, 2000.

6. Atkins, P.W., *Physical Chemistry*. 6th ed. Freeman, New York, 1998.

7. Ando, S., Sakamoto, E. and Iba, H., "Evolutionary modeling and inference of gene network," *Information Sciences*, vol. 145, no. 3-4, pp. 237-259, 2002.

8. Au, W.H., Chan, K.C.C. and Yao, X., "A novel evolutionary data mining algorithm with applications to churn prediction," *IEEE Trans. Evolutionary Computation*, Special Issue on Data Mining and Knowledge Discovery with Evolutionary Algorithms, vol. 7, no. 6, pp. 532-545, 2003.

9. Arima, C., Taizo, H. and Okamoto, M., “Gene expression analysis using fuzzy k-means clustering,” *Genome Informatics*, vol. 14, pp. 334-335, 2003.

10. Baldi, P. and Brunak, S., *Bioinformatics: The Machine Learning Approach*. MIT Press, Cambridge, MA, 2001.

11. Baxevanis, A.D. and Ouellette, B.F.F, *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins*. John Wiley & Sons, Inc., 2005.

12. Borodovsky, M. and McIninch, J., “GeneMark: parallel gene recognition for both DNA strands,” *Comput. Chem.*, vol. 17, pp. 123-133, 1993.

13. Burge, C.B. and Karlin, S., “Finding the genes in genomic DNA,” *Curr. Opin. Struct. Biol.*, vol. 8, pp. 346-354, 1998.

14. Bourne, P.E. and Weissig, H., *Structural Bioinformatics*. Wiley-Liss, 2003.

15. Bower, J.M. and Bolouri, H., *Computation Modeling of Genetic and Biochemical Networks*. Cambridge, Mass.: MIT Press, 2001.

16. Berrar, D.P., Dubitzky, W. and Granzow, M., *A Practical Approach to Microarray Data Analysis*. Boston, MA: Kluwer Acad., 2003.

17. Ben-Dor, A., Shamir, R. and Yakhini, Z., “Clustering gene expression patterns,” *J. Comp. Biol.*, vol. 6, no. 3-4, pp. 281-297, 1999.

18. Balazsi, G., Kay, K.A., Barabasi, A.L. and Oltvai, Z.N., "Spurious spatial periodicity of co-expression in microarray data due to printing design," *Nucleic Acids Res.*, vol. 31, no. 15, pp. 4425-4433, 2003.

19. Ball, C.A., Jin, H., Sherlock, G., Weng, S., Matese, J.C., Andrada, R., Binkley, G., Dolinski, K., Dwight, S.S., Harris, M.A., Issel-Tarver, L., Schroeder, M., Botstein, D. and Cherry, J.M., “Saccharomyces genome database provides tools to survey gene expression and functional analysis data,” *Nucleic Acids Res.*, vol. 29, no. 1, pp. 80-81, 2001.

20. Bezdek, J.C., *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, New York, 1981.

21. Babu, G.P. and Murty, M.N., "Clustering with evolution strategies," *Pattern Recognition*, vol. 27, no. 2, pp. 321 - 329, 1994.

22. Baeck, T., Fogel, D. and Michalewicz, Z., *Handbook of Evolutionary Computation*. U.K.: Institute of Physics, 1997.

23. Baeck, T., Fogel, D. and Michalewicz, Z., *Evolutionary Computation 1: Basic Algorithms and Operators*. U.K.: Institute of Physics, 2000.

24. Baeck, T., Fogel, D. and Michalewicz, Z., *Evolutionary Computation 2: Advanced Algorithms and Operators*. U.K.: Institute of Physics, 2000.

25. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N. and Bourne, P.E., "The protein data bank," *Nucleic Acids Res.*, vol. 28, pp. 235-242, 2000.

26. Collins, F.S., Morgan, M. and Patrinos, A., “The human genome project: lessons from large-scale biology,” *Science*, vol. 300, no. 5617, pp. 286-290, 2003.

27. Cox, T.M. and Sinclair, J., *Molecular Biology in Medicine*. Oxford; Cambridge, Mass.: Blackwell Science, 1997.

28. Cios, K.J., Pedrycz, W. and Swiniarski, R., *Data Mining Methods for Knowledge Discovery*. Dordrecht: Kluwer, 1998.

29. Cho, R.J., Campbell, M.J., Winzeler, E.A., Steinmetz, L., Conway, A., Wodicka, L., Wolfsberg, T.G., Gabrielian, A.E., Landsman, D., Lockhart, D.J. and Davis, R.W., "A genome-wide transcriptional analysis of the mitotic cell cycle," *Mol. Cell*, vol. 2, no. 1, pp. 65-73, 1998.

30. Cho, R.J., Huang, M., Campbell, M.J., Dong, H., Steinmetz, L., Sapinoso, L., Hampton, G., Elledge, S.J., Davis, R.W. and Lockhart, D.J., "Transcriptional regulation and function during the human cell cycle," *Nat. Genet.*, vol. 27, no. 1, pp. 48-54, 2001.

31. Chen, K.C., Wang, T.Y., Tseng, H.H., Huang, C.Y. and Kao, C.Y., "A stochastic differential equation model for quantifying transcriptional regulatory network in Saccharomyces cerevisiae," *Bioinformatics*, vol. 21, no.12, pp. 2883-2890, 2005.

32. Chen, Y.P.P., *Bioinformatics Technologies*. Springer, 2005.

33. Chickering, D.M., *Learning Bayesian Networks from Data*. UCLA Cognitive Systems Laboratory, Technical Report, R-245, 1996.

34. Chan, K.C.C. and Wong, A.K.C., "A statistical technique for extracting classificatory knowledge from databases," *Knowledge Discovery in Databases*, G. Piatesky-Shapiro and W.J. Frawley, Eds. Menlo Park, CA:/Cambridge, MA: AAAI/MIT Press, pp. 107-123, 1991.

35. Ching, J.Y., Wong, A.K.C. and Chan, K.C.C., "Class-dependent discretization for inductive learning from continuous and mixed-mode data," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, no. 7, pp. 641-651, 1995.

36. Chan, K.C.C., Wong, A.K.C. and Chiu, D.K.Y., "Learning sequential patterns for probabilistic inductive prediction," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 24, no. 10, pp. 1532-1547, 1994.

37. Corne, D. and Pridgeon, C., "Investigating issues in the reconstructability of genetic regulatory networks," *Proc. of the IEEE Congress on Evolutionary Computation*, pp. 582-589, 2004.

38. Chu, S., DeRisi, J., Eisen, M., Mulholland, J., Botstein, D., Brown, P.O. and Herskowitz, I., "The transcriptional program of sporulation in budding yeast," *Science*, vol. 282, no. 5389, pp. 699-705, 1998.

39. Chi, Z., Weimin, X., Tirpak, T.M. and Nelson, P.C., "Evolving accurate and compact classification rules with gene expression programming," *IEEE Trans. Evolutionary Computation*, vol. 7, no. 6, pp. 519-531, 2003.

40. Cano, J.R., Herrera, F. and Lozano, M., "Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study," *IEEE Trans. Evolutionary Computation*, vol. 7, no. 6, pp. 561-575, 2003.

41. Chan, K.C.C. and Chung, L.L.H., "Discovering clusters in databases containing mixed continuous and discrete-valued attributes," *in Proc. of SPIE AeroSense'99 Data Mining and Knowledge Discovery: Theory, Tools, and Technology*, pp. 22-31, 1999.

42. Chevalier, S. and Blow, J.J., "Cell cycle control of replication initiation in eukaryotes," *Curr. Opin. Cell Biol.*, vol. 8, pp. 815-821, 1996.

43. Chung, K.F.L., *Lecture Notes: Data Mining and Data Warehouse*, 2003.

44. Durbin, R., Eddy, S.R., Krogh, A. and Mitchison, G.J., *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

45. DeRisi, J.L., Iyer, V.R., and Brown, P.O., "Exploring the metabolic and genetic control of gene expression on a genomic scale," *Science*, vol. 278, pp. 680-686, 1997.

46. Dejori, M. and Stetter, M., "Bayesian inference of genetic networks from gene expression data: convergence and reliability," *Proc. of the International Conference on Artificial Intelligence*, pp. 321-327, 2003.

47. Davies, D.L. and Bouldin, D.W., "A cluster separation measure," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 1, no. 2, pp. 224-227, 1979.

48. De la Fuente, A., Brazhnik, P. and Mendes, P., "Linking the genes: inferring quantitative gene networks from microarray data," *Trends Genet.*, vol. 18, no. 8, pp. 395-398, 2002.

49. Ding, C., "Analysis of gene expression profiles: class discovery and leaf ordering," *in Proc. of the International Conference of Research in Computational Molecular Biology (RECOMB)*, pp. 127-136, 2002.

50. DeGroot, M.H. and Schervish, M.J., *Probability and Statistics*, Boston: Addison-Wesley, 2002.

51. Ewens, W.J. and Grant, G.R., *Statistical Methods in Bioinformatics: An Introduction*. Springer-Verlag, New York, 2001.

52. 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*, vol. 95, no. 25, pp. 14863-14868, 1998.

53. Endy, D. and Brent, R., "Modeling cellular behaviour," *Nature*, vol. 409, pp. 391-395, 2001.

54. Eidhammer, I., *Protein Bioinformatics: An Algorithmic Approach to Sequence and Structure Analysis*. John Wiley & Sons, 2004.

55. Falkenauer, E., *Setting New Limits in Bin Packing with a Grouping GA Using Reduction*. Technical Report, CRIF Industrial Automation, Brussels, 1994.

56. Falkenauer, E., "A hybrid grouping genetic algorithm for bin packing," *J. of Heuristics*, vol. 2, no. 1, pp. 5-30, 1996.

57. Fernandes, P.M., Domitrovic, T., Kao, C.M. and Kurtenbach, E., "Genomic expression pattern in Saccharomyces cerevisiae cells in response to high hydrostatic pressure," *FEBS Letter*, vol. 556, no. 1-3, pp. 153-160, 2004.

58. Futschik, M.E. and Kasabov, N.K., "Fuzzy clustering of gene expression data," *Proc. of the World Congress of Computational Intelligence WCCI 2002*, IEEE Press, 2002.

59. Franti, P., Kivijarvi, J., Kaukoranta, T. and Nevalainen, O., "Genetic algorithms for large-scale clustering problems," *The Computer Journal*, vol. 40, no. 9, pp. 547-554, 1997.

60. Falkenauer, E., *Genetic Algorithms and Grouping Problems*. Chichester; New York: Wiley, 1998.

61. Filkov, V., Skiena, S. and Zhi, J., "Analysis techniques for microarray time-series data," *in Proc. of the International Conference of Research in Computational Molecular Biology (RECOMB)*, 2001.

62. Farabee, M., *Online Biology Book*. Available: http://www.estrellamountain.edu/faculty/farabee/biobk/biobooktoc.html

63. Gasch, A.P. and Eisen, M.B., "Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering," *Genome Biol.*, vol. 3, no. 11, pp. 1-22, 2002.

64. Goldberg, D.E., *Genetic Algorithms in Search, Optimization and Machine Learning*. New York: Addison-Wesley, 1989.

65. Golub, T.R., "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, pp. 531-537, 1999.

66. Goransson, O., *Market Basket Analysis*. Available: http://www.megaputer.com/products/pa/algorithms/ba.php3

67. Giaever, G., Shoemaker, D.D., Jones, T.W., Liang, H., Winzeler, E.A., Astromoff, A. and Davis, R. W., "Genomic profiling of drug sensitivities via induced haploinsufficiency," *Nat. Genet.*, vol. 21, no. 3, pp. 278-283, 1999.

68. Han, J. and Kamber, M., *Data Mining: Concepts and Techniques*. San Francisco: Morgan Kaufmann Publishers, 2001.

69. Handl, J. and Knowles, J., "An evolutionary approach to multiobjective clustering," *IEEE Trans. on Evolutionary Computation*. In Press, 2006.

70. Hastie, T., Tibshirani, R. and Friedman, J., *The Elements of Statistical Learning: Data Mining, Inference and Prediction.* Springer-Verlag, New York, 2001.

71. Higgins, D.G. and Sharp, P.M., "Clustal: a package for performing multiple sequence alignment on a microcomputer," *Gene*, vol. 73, pp. 237-244, 1988.

72. Huang, X. and Madan, A., "CAP3: a DNA sequence assembly program," *Genome Research*, vol. 9, pp. 868-877, 1999.

73. Hartl, D.L. and Jones, E.W., *Genetics: Analysis of Genes and Genomes*. Sudbury, MA: Jones & Bartlett, 2001.

74. Herrero, J., Valencia, A. and Dopazo, J., "A hierarchical unsupervised growing neural network for clustering gene expression patterns," *Bioinformatics*, vol. 17, no. 2, pp. 126-136, 2001.

75. Husmeier, D., "Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic bayesian networks," *Bioinformatics*, vol. 19, no. 17, pp. 2271-2282, 2003.

76. Heckerman, D., Geiger, D. and Chickering, D.M., "Learning bayesian networks: the combination of knowledge and statistical data," *Machine Learning*, vol. 20, pp. 197-243, 1995.

77. Haberman, S.J., "The analysis of residuals in cross-classified tables," *Biometrics*, vol. 29, pp. 205-220, 1973.

78. Helden, J.V., Andre, B. and Collado-Vides, J., "Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies," *J. Mol Biol.*, vol. 281, no. 5, pp. 827-842, 1998.

79. Helden, J.V., Rios, A.F. and Collado-Vides, J., "Discovering regulatory elements in non-coding sequences by analysis of spaced dyads," *Nucleic Acids Res.*, vol. 28, no. 8, pp. 1808-1818, 2000.

80. Hall, L.O., Ozyurt, I.B. and Bezdek, J.C., "Clustering with a genetically optimized approach," *IEEE Trans. Evolutionary Computation*, vol. 3 no. 2, pp. 103-112, 1999.

81. Holm, L. and Sander, C., "Protein structure comparison by alignment of distance matrices," *J. Mol. Biol.*, vol. 233, pp. 123-138, 1993.

82. Hogue, C.W.V., Ohkawa, H. and Bryant, S.H., "A dynamic look at structures: WWW-entrez and the molecular modeling database," *Trends in Biochemical Sci.*, vol. 21, pp. 226-229, 1996.

83. Iyer, V.R., Eisen, M.B., Ross, D.T., Schuler, G., Moore, T., Lee, J.C.F., Trent, M.J., Staudt, M.L., Hudson Jr, J., Bogosk, M.S. *et al.*, "The transcriptional program in the response of human fibroblast to serum," *Science*, vol. 283, pp. 83-87, 1999.

84. Jain, A.K. and Dubes, R.C., *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, NJ, 1998.

85. Kohonen, T., *Self-Organization and Associative Memory*. New York: Springer-Verlag, 1989.

86. Kowalski, G., *Information Retrieval Systems - Theory and Implementation*. Kluwer Academic Publishers, 1997.

87. Krishna, K. and Murty, M.N., "Genetic k-means algorithm," *IEEE Trans. on Systems, Man and Cybernetics-Part B*, vol. 29, no. 3, pp. 433-439, 1999.

88. Kanehisa, M. and Goto, S., "KEGG: kyoto encyclopedia of genes and genomes," *Nucleic Acids Res.*, vol. 28, pp. 27-30, 2000.

89. Karp, P.D., et al., "The MetaCyc database," *Nucleic Acids Res.*, vol. 30, no. 1, pp. 59-61, 2002.

90. Lockhart, D.J. and Winzeler, E.A., "Genomics, gene expression and DNA arrays," *Nature*, vol. 405, no. 6788, pp. 827-836, 2000.

91. Lesk, A.M., *Introduction to Bioinformatics*. Oxford University Press, New York, 2002.

92. Lapointe, J., Li, C., Higgins, J.P., Van De Rijn, M., Bair, E., Montgomery, K., Ferrari, M., Egevad, L., Rayford, W., Bergerheim, U., Ekman, P., DeMarzo, A.M., Tibshirani, R., Botstein, D., Brown, P.O., Brooks, J.D. and Pollack, J.R., "Gene expression profiling identifies clinically relevant subtypes of prostate cancer," *Proc. Natl. Acad. Sci. USA*, vol. 101, no. 3, pp. 811-816, 2004.

93. Lashkari, D.A., DeRisi, J.L., McCusker, J.H., Namath, A.F., Gentile, C., Hwang, S.Y., Brown, P.O. and Davis, R.W., "Yeast microarrays for genome wide parallel genetic and gene expression analysis," *Proc. Natl. Acad. Sci. USA*, vol. 94, no. 24, pp. 13057-13062, 1997.

94. Leloup, J.C. and Goldbeter, A., "Toward a detailed computational model for the mammalian circadian clock," *Proc. of the National Academy of Science, USA*, vol. 100, pp. 7051-7056, 2003.

95. Lu, Y. and Han, J., "Cancer classification using gene expression data," *Information Systems*, vol. 28, no. 4, pp. 243-268, 2003.

96. Larsen, B. and Aone, C., "Fast and effective text mining using linear-time document clustering," *Proc. of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.16-22, 1999.

97. Liu, C.L., *Introduction to Combinatorial Mathematics*. New York: McGraw-Hill, 1968.

98. Mount, D.W., *Bioinformatics: Sequences and Genome Analysis*. Cold Spring Harbor, New York, 2004.

99. Mehmed, K., *Data Mining: Concepts, Models, Methods, and Algorithms*. Hoboken, N.J.: Wiley-Interscience: IEEE Press, 2003.

100. MacQueen, J., "Some methods for classification and analysis of multivariate observation," *Proc. Symp.Math. Stat. and Prob. Berkeley*, vol. 1, pp. 281-297, 1967.

101. Mitra, S. and Acharya, T., *Data Mining: Multimedia, Soft Computing, and Bioinformatics*. John Wiley & Sons, 2003.

102. Middendorf, M., Kundaje, A., Wiggins, C., Freund, Y. and Leslie, C., "Predicting genetic regulatory response using classification," *Bioinformatics*, vol. 20 (Suppl. 1), pp. i232-i240, 2004.

103. Mateos, A., Dopazo, J., Jansen, R., Tu, Y., Gerstein, M. and Stolovitzky, G., "Systematic learning of gene functional classes from DNA array expression data by using multiplayer perceptrons," *Genome Res.*, vol. 12, no. 11, pp. 1703-1715, 2002.

104. Mewes, H.W., Frishman, D., Guldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Munsterkotter, M., Rudd, S. and Weil, B., "MIPS: a database for genomes and protein sequences," *Nucleic Acids Res.*, vol. 30, pp. 31-34, 2002.

105. Muni, D.P., Pal, N.R. and Das, J., "A novel approach to design classifiers using genetic programming," *IEEE Trans. Evolutionary Computation*, vol. 8, no. 2, pp. 183-196, 2004.

106. Murthy, C.A. and Chowdhury, N., "In search of optimal clusters using genetic algorithms," *Pattern Recognition Letters*, vol. 17, no. 8, pp. 825-832, 1996.

107. Maulik, U. and Bandyopadhyay, S., "Genetic algorithm-based clustering technique," *Pattern Recognition*, vol. 33, pp. 1455 - 1465, 2000.

108. Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*. New York: Springer Verlag, 1996.

109. Matoba, R., Kato, K., Kurooka, C., Maruyama, C., Sakakibara, Y. and Matsubara, K., "Correlation between gene functions and developmental

expression patterns in the mouse cerebellum," *Eur. J. Neurosci.*, vol. 12, no. 4, pp. 1357-1371, 2000.

110. Murzin, A.G., Brenner, S.E., Hubbard, T. and Chothia C., "SCOP: a structural classification of proteins database for the investigation of sequences and structures," *J. Mol. Biol.*, vol. 247, pp. 536-540, 1995.

111. Ma, P.C.H., Chan, K.C.C., Yao, X. and Chiu, D.K.Y., "An evolutionary clustering algorithm for gene expression microarray data analysis," accepted by *IEEE Trans. on Evolutionary Computation*, to appear in 2006.

112. Ma, P.C.H., Chan, K.C.C. and Chiu, D.K.Y., "Clustering and re-clustering for pattern discovery in gene expression data," *Journal of Bioinformatics and Computational Biology*, vol. 3, no. 2, pp. 281-301, 2005.

113. Ma, P.C.H. and Chan, K.C.C., "A novel data mining algorithm for reconstructing gene regulatory networks from microarray data," *Proc. of the 21st Annual ACM Symposium on Applied Computing (ACMSAC06 – Bioinformatics)*, Dijon, France, Apr. 23-27, 2006, pp. 202-203.

114. Ma, P.C.H. and Chan, K.C.C., "An effective data mining technique for the discovery of gene regulatory networks from time-series expression data," Submitted for journal publication (under review).

115. Ma, P.C.H. and Chan, K.C.C., "Inference of gene regulatory networks from microarray data," *Proc. of the Fourth Asia Pacific Bioinformatics Conference (APBC06)*, Taipei, Taiwan, Feb. 13-16, 2006, pp. 17-26. Also, in the series on *Advances in Bioinformatics & Computational Biology*, vol. 3, Imperial College Press, 2006.

116. Ma, P.C.H. and Chan, K.C.C., "A clustering algorithm for discovering overlapping clusters from noisy gene expression data," Submitted for journal publication (under review).

117. Ma, P.C.H. and Chan, K.C.C., *Mining Overlapping Clusters in Gene Expression Data*. Technical Report, COMP-02-05, 2005, Department of Computing, The Hong Kong Polytechnic University.

118. Ma, P.C.H. and Chan, K.C.C., *Discovering Clusters in Gene Expression Data*. Technical Report, COMP-07-04, 2004, Department of Computing, The Hong Kong Polytechnic University.

119. Ma, P.C.H. and Chan, K.C.C., "Discovering clusters in gene expression data using evolutionary approach," *Proc. of the 15th IEEE International Conf. on Tools with Artificial Intelligence (ICTAI03)*, Sacramento, California, USA, Nov. 3-5, 2003, pp.459-466.

120. Ma, P.C.H., Chan, K.C.C. and Chiu, D.K.Y., "Discovering clusters in gene expression data," *Proc. of the 7th Joint Conf. on Information Sciences (jointly 5th Atlantic Symposium on Computational Biology and Genome Informatics,*

CBGI03), Research Triangle Park, N. Carolina, USA, Sept. 26-30, 2003, pp.879-882.

121. Ma, P.C.H. and Chan, K.C.C., "Clustering gene expression data with hybrid GA approach," *Proc. of the 7th IASTED International Conf. on Artificial Intelligence and Soft Computing (ASC03)*, Banff, Alberta, Canada, Jul. 14-16, 2003, pp.223-228. Also in *Artificial Intelligence and Soft Computing*, ACTA Press, 2003.

122. Naraghi, M. and Neher, E., "Linearized buffered Ca^{2+} diffusion in microdomains and its implications for calculation of $[Ca^{2+}]$ at the mouth of a calcium channel," *J. Neurosci.*, vol. 17, pp. 6961-6973, 1997.

123. Nadia, B., Azuaje, F. and Cunningham, P., "An integrated tool for microarray data clustering and cluster validity assessment," *Bioinformatics*, vol. 21, pp. 451-455, 2005.

124. Osteyee, D.B. and Good, I.J., *Information, Weight of Evidence, the Singularity between Probability Measures and Signal Detection*. Berlin: Springer-Verlag, 1974.

125. Orengo, C.A., Michie, A.D., Jones, S., Jones, D.T., Swindells, M.B. and Thornton, J.M., "CATH - A hierarchic classification of protein domain structures," *Structure*, vol. 5, no. 8, pp. 1093-1108, 1997.

126. Perrin, B.E., Ralaivola, L., Mazurie, A., Bottani, S., Mallet, J. and Buc, F., "Gene networks inference using dynamic bayesian networks," *Bioinformatics*, vol. 19 (Suppl. 2), pp. ii138-ii148, 2003.

127. Pham, D. T., *Intelligent Optimisation Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks*, London; New York: Springer, 2000.

128. Park, Y. and Song, M., "A genetic algorithm for clustering problems," *Genetic Programming 1998: Proc. 3rd Annual Conf.*, pp. 568-575, Morgan Kaufmann Publishers, 1998.

129. Quinlan, J.R., "Induction on decision trees," *Machine Learning*, vol. 1, pp. 81-106, 1986.

130. Quackenbush, J., "Computational analysis of microarray data," *Nat. Rev. Genet.*, vol. 2, no. 6, pp. 418-427, 2001.

131. Quinlan, J.R., *C4.5: Programs for Machine Learning*. San Fran., CA: Morgan Kaufmann, 1993.

132. Ramsay, G., "DNA chips - states-of-the-art," *Nature Biotechnology*, vol. 16, no. 1, pp. 40-44, 1998.

133. Rabiner, L.R., “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proc. IEEE*, vol. 77, pp. 257-286, 1989.

134. Raeymaekers, L., “Dynamics of boolean networks controlled by biologically meaningful functions,” *J. of Theoretical Biology*, vol. 218, pp. 331-341, 2002.

135. Rousseeuw, J.P., “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis,” *J. Comp. Appl. Math.*, vol. 20, pp. 53-65, 1987.

136. Schena, M., Shalon, D., Davis, R.W. and Brown, P.O., “Quantitative monitoring of gene expression patterns with a complementary DNA microarray,” *Science*, vol. 270, no. 5235, pp. 467-470, 1995.

137. Stein, L.D., "Human genome: end of the beginning," *Nature*, vol. 431, pp. 915-916, 2004.

138. Silberschatz, A. and Tuzhilin, A., “What makes patterns interesting in knowledge discovery systems,” *IEEE Trans. on Knowledge and Data Engineering*, vol. 8, pp. 970-974, 1996.

139. Smith, T.F. and Waterman, M.S., “Comparison of biosequences,” *Adv. Appl. Math.*, vol. 2, pp. 482-489, 1981.

140. Salzberg, S., Delcher, A., Kasif, S. and White, O., “Microbial gene identification using interpolated Markov models,” *Nucleic Acids Research*, vol. 26, pp. 544-548, 1998.

141. Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D. and Futcher, B., “Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization,” *Mol. Biol. Cell*, vol. 9, no. 12, pp. 3273-3297, 1998.

142. Seiffert, U., *Bioinformatics Using Computational Intelligence Paradigms*. Springer, 2005.

143. Stekel, D., *Microarray Bioinformatics*. Cambridge University Press, 2003.

144. Schena, M., *DNA Microarrays: A Practical Approach*. Oxford; N.Y.: Oxford University Press, 1999.

145. Suzuki, M., Brenner, S.E., Gerstein, M. and Yagi, N., “DNA recognition code of transcription factors,” *Protein Eng.*, vol. 8, no. 4, pp. 319-328, 1995.

146. Su, Y., Murali, T.M., Pavlovic, V., Schaffer, M. and Kasif, S., "RankGene: Identification of diagnostic genes based on expression data," *Bioinformatics*, vol. 19, no. 12, pp. 1578-1579, 2003.

147. Tou, J.T. and Gonzalez, R.C., *Pattern Recognition Principles*. London: Addison-Wesley, 1974.

148. Tavazoie, S., Hughes, J.D., Campbell, M.J., Cho, R.J. and Church, G.M., “Systematic determination of genetic network architecture,” *Nat. Genet.*, vol. 22, no. 3, pp. 281-285, 1999.

149. Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E.S. and Golub, T.R., “Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation,” *Proc. Natl. Acad. Sci. USA*, vol. 96, no. 6, pp. 2907-2912, 1999.

150. Tyson, J.J., “Models of cell cycle control in eukaryotes,” *J. of Biotechnology*, vol. 71, pp. 239-244, 1999.

151. Tu, Y., Stolovitzky, G. and Klein, U., "Quantitative noise analysis for gene expression microarray experiments," *Proc. Natl. Acad. Sci. USA*, vol. 99, no. 22, pp. 14031-14036, 2002.

152. Van Berkum, N.L. and Holstege, F.C., “DNA microarrays: raising the profile,” *Curr. Opin. Biotechnol.*, vol. 12, no. 1, pp. 48-52, 2001.

153. Wilson, C.L., Pepper, S.D., Hey, Y. and Miller, C.J., "Amplification protocols introduce systematic but reproducible errors into gene expression studies," *BioTechniques*, vol. 36, no. 3, pp. 498-506, 2004.

154. Weaver, R.F., *Molecular Biology*. Boston: McGraw-Hill, 2005.

155. Wang, J.T.L., *Data Mining in Bioinformatics*. Springer, 2005.

156. Ward, J.H., “Hierarchical grouping to optimize an objective function,” *J. Am. Stat. Assoc.*, vol. 58, pp. 236-244, 1963.

157. Watson, J.D., *Molecular Biology of the Gene*. San Francisco, Calif.: Pearson/Benjamin Cummings, 2004.

158. Wang, Y. and Wong, A.K.C., "From association to classification: inference using weight of evidence," *IEEE Trans. Knowledge and Data Engineering*, vol. 15, no. 3, pp. 764-767, 2003.

159. Witten, I.H. and Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques*. 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

160. Xiong, M., Fang, X. and Zhao, J., “Biomarker identification by feature wrappers,” *Genome Res.*, vol. 11, pp. 1878-1887, 2001.

161. Yeung, K.Y. and Ruzzo, W.L., “Principal component analysis for clustering gene expression data,” *Bioinformatics*, vol. 17, no. 9, pp. 763-774, 2001.

162. Zaïane, O.R., *Introduction to Data Mining*. Available: http://www.cs.ualberta.ca/~zaiane

163. Zheng, J., Wu, J. and Sun, Z., “An approach to identify over-represented cis-elements in related sequences,” *Nucleic Acids Res.*, vol. 31, no. 7, pp. 1995-2005, 2003.

164. Zou, M. and Conzen, S.D., “A new dynamic bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data,” *Bioinformatics*, vol. 21, no. 1, pp. 71-79, 2005.

165. Zell, A., Spieth, C., Streichert, F. and Speer, N., "Multi-objective model optimization for inferring gene regulatory networks," *Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, pp. 607-620, 2005.