Extracting Gene Networks - Rule and Pattern based approaches

Tara Elizabeth McIntosh

Bachelor of Science (Bioinformatics) (Honours)
Supervisor: Dr Sanjay Chawla
The University of Sydney

16 June, 2005

Abstract

Microarrays are revolutionising the way in which biological experiments are carried out. They allow thousands of genes to be studied, unlike traditional wet-lab experiments. A main aim of molecular biology is to understand how the functions of genes relate to each other. These relationships can be depicted in gene networks.

Existing data mining techniques for extracting gene networks from microarray data have suffered from two major weaknesses – limiting the number of genes and the number of consecutive time frames that can be incorporated into the construction process. This is because microarray data is dense rendering many data mining techniques infeasible.

A new family of top-down association rule mining algorithms have emerged to facilitate mining in dense datasets. However, until now, all the algorithms proposed rely on the support measure to prune the search space. This is a major shortcoming as it results in the pruning of *many potentially interesting* relationships which have low support and high confidence.

In this thesis we provide the first comprehensive solution to globally mine microarray data in a top-down manner without the support-threshold. An evaluation of our algorithm MAXCONF against an existing support pruning method, shows that we increase recall from 0.15% to 94%.

We also propose the first top-down algorithm (SEQRE) for sequential pattern mining. SEQRE efficiently mines sequential patterns from temporal data considering all consecutive time frames and all genes. Our approach identifies many biological relationships and is the first algorithm which has been able to handle the complexity of the temporal microarray datasets. Finally we will synthesise and visualise gene networks based on the relationships discovered with MAXCONF and SEQRE.

We provide complete, principled and efficient solutions for the mining, integration and analysis of microarrays.

Acknowledgements

To my family and Lang. Thank you for your love, understanding and endless patience and encouragement. Thank you for always being there when I have needed you most.

I would like to gratefully acknowledge the enthusiastic supervision of Dr. Sanjay Chawla during this work.

To James Curran, who has dedicated many hours of his time to me throughout the past year. Thank you for your continual support and motivating talks.

To Mark Assad. Thank you for your friendship and help this past year. Thank you for being an *unreal* friend.

Finally, thank you to all previous and current Honours students and friends. Thank you for making honours not just a year of study and stress.

Contents

1 Introduction **1**

1.1 Motivation and Aims . . . 2

1.2 Contributions . . . 3

1.3 Thesis Overview . . . 5

2 Molecular Biology **6**

2.1 Yeast . . . 6

2.2 Gene Expression . . . 7

2.3 Gene Networks . . . 8

2.4 Microarrays . . . 10

2.4.1 Knockout Experiments . . . 10

2.4.2 Temporal Experiments . . . 11

2.4.3 Discretisation . . . 12

2.4.4 Computational Issues . . . 12

2.5 Summary . . . 13

3 Learning problems associated with Gene Networks **14**

3.1 Clustering . . . 14

3.1.1 Hierarchical Clustering . . . 15

3.1.2 Significance of Clusters . . . 16

3.2 Boolean Networks . . . 16

3.2.1 Noisy Boolean Networks . . . 19

3.2.2 Probabilistic Boolean Networks . . . 19

3.2.3 Temporal Boolean Networks . . . 19

3.3 Bayesian Networks 20
3.4 Summary 21

4 Association Rule Mining **22**
4.1 Frequent Itemsets and Association Rules 22
4.2 Itemset Mining - step 1 24
4.2.1 Search Space 24
4.2.2 Closed Itemsets 25
4.3 Frequent Itemset Algorithms 26
4.3.1 The Apriori Algorithm 27
4.3.1.1 Application to Microarray Data 28
4.3.2 Row Enumeration 29
4.3.2.1 Motivation 29
4.3.2.2 RERII 30
4.4 Summary 33

5 Advances for Gene Networks from Knockout data **38**
5.1 Bottom-up High Confidence Rule Mining 38
5.2 MAXCONF 40
5.2.1 Level 1 Confidence Pruning 41
5.2.2 Level 2 Confidence Pruning 42
5.2.3 Maximal Confident Rules 43
5.3 MaxConf Example 44
5.4 MAXCONF Evaluation 45
5.4.1 Computational Effectiveness 47
5.4.2 Biological Rule Evaluation 50
5.4.3 BIND 51
5.4.3.1 Direct Interactions 51
5.4.3.2 Indirect Interactions 53
5.4.4 GOstat 54
5.4.4.1 Iron Uptake Pathway 55
5.5 Summary 56

6 Advances for Gene Networks from Temporal data **59**

6.1 Sequential Pattern Mining 60
6.1.1 Search Space 61
6.2 SEQRE 62
6.2.1 SEQRE Algorithm 63
6.2.1.1 Tree Representation 65
6.2.1.2 Child Node Construction 65
6.2.1.3 Maximal Sequential Pattern Pruning 66
6.3 SEQRE Analysis 68
6.3.1 Computational Effectiveness 68
6.3.2 Biological Analysis 73
6.4 Gene Network Construction 74
6.4.1 Association Rule Network 75
6.4.2 Local Gene Network 76
6.5 Summary 79

7 Conclusions and Future Work **84**
7.1 Conclusions 84
7.2 Future Work 85

Bibliography **87**

List of Figures

2.1 Genetic information flow in a cell 8
2.2 Yeast cell cycle gene network 9

3.1 Hierarchical tree (LMIL, 2005) 15
3.2 Simple boolean network 18

4.1 Apriori algorithm example (Min Support = 2) 28
4.2 RERII algorithm example - Part 1 (Min Support ≥ 3) 35
4.3 RERII algorithm example - Part 2 (Min Support ≥ 3) 36
4.4 RERII algorithm example - Part 3(Min Support ≥ 3) 37

5.1 MAXCONF tree - no pruning 46
5.2 MAXCONF tree - closure pruning 46
5.3 MAXCONF tree - confidence pruning 47
5.4 Scalability of confidence and support pruning methods 49
5.5 Rules discovered with confidence and support pruning methods . 49

6.1 RERII with sequential transactions in Table 6.1 63
6.2 Sequential-item-enumeration tree 64
6.3 SEQRE algorithm example - Part 1 (Min Support = 2) 69
6.4 SEQRE algorithm example - Part 2 (Min Support = 2) 70
6.5 SEQRE algorithm example - Part 3 (Min Support = 2) 71
6.6 Scalability of SEQRE 72
6.7 Example ARN and LGN 77
6.8 LGN with goal gene CLN3 where distance = 3 79
6.9 LGN with goal gene STE2 where distance = 3 79

6.10 LGN with goal gene POL30 where distance = 3 80
6.11 Sequential LGN with goal gene CDC5 where distance = 3 80

List of Tables

3.1 Boolean expression table 17

4.1 Example dataset 26
4.2 Frequent Closed Itemsets 32

5.1 Example transaction set 44
5.2 Rules identified by MAXCONF and RERII 45
5.3 Effect of confidence pruning for rule extraction 48
5.4 Range of rule supports 48
5.5 Direct interactions identified in rules 52
5.6 Association rules exhibiting direct interactions in yeast 53
5.7 Association rules exhibiting indirect interactions in yeast 53
5.8 GO clusters identified in rules 54
5.9 Association rules related to the Iron Uptake Pathway in yeast . . . 55
5.10 Example rules containing GO clusters 58

6.1 Example sequential transaction set 63
6.2 Effect of support closure pruning for pattern extraction 72
6.3 Robustness of SEQRE for pattern extraction 73
6.4 Example patterns extracted by SEQRE 73
6.5 Example association rules 76

List of Algorithms

1 Apriori - itemset mining 29
2 RERII - closed itemset mining - Part 1 33
3 RERII - closed itemset mining - Part 2 34
4 MAXCONF - highly confident itemset mining 57
5 SEQRE - maximal sequential pattern mining - Part 1 81
6 SEQRE - maximal sequential pattern mining - Part 2 82
7 Local Gene Network 83

Chapter 1

Introduction

The increasing volume of biological data collected in recent years requires the development of efficient bioinformatic tools for genomic and proteomic data analysis. The *microarray* is revolutionary in the biological domain as it allows one to study the behaviour of all the genes within a cell in only one experiment. However most genes known to be involved in a particular process are still identified in painstaking wet-lab experiments which allow only a few genes to be studied at a time. This is not due to the problems in microarray experiments, but in the analysis and interpretation of the large volumes of data.

One main objective of biologists is to develop a deeper understanding of the different mechanisms by which different cells control and regulate the transcription of their genes. More specifically, to extract gene regulatory networks from microarray data.

This thesis presents *association rule mining* and *sequential pattern mining* approaches to extracting gene networks from two distinct forms of microarray data. This chapter outlines the motivation and aims, along with the research contributions, of this project.

1.1 Motivation and Aims

Bioinformatics is providing researchers with analysis methods not considered possible until recently. The information gathered from bioinformatic analysis provides insight into the unknown data, suggesting new hypotheses directing researchers to design appropriate or more precise experiments to solve their specific problems. This is extremely valuable as prior knowledge often reduces the number of tedious time consuming lab experiments required. For example, knowing a protein's predicted function, experiments can be designed precisely around this, leading to a result much more effectively. In comparison, if no information is known in advance many trial and error experiments are likely to be conducted before any concrete research direction is identified.

The functions of many genes and their relationships with other genes remain unknown. Further with the advances in experimental techniques, many new discoveries are made with genes whose function and relationships were once considered understood.

Extracting gene relationships within an organisms is traditionally a tedious task, and will continue to be if the wealth of information hidden within microarray data is not analysed effectively. Microarray data contains the expression levels of thousands of genes (N) in M various experiments, and is often expressed as a N x M matrix.

Existing methods for predicting gene networks are limited by the number of genes and consecutive time points that can be analysed. This issue is a direct consequence of the existing algorithms which are exponential with respect to the number of genes and times studied. Considering the number of genes on a single microarray is in the thousands if not tens of thousands, it is unacceptable to limit the genes investigated. In addition, temporal expression data reveals many important gene relationships. Furthermore with the cost of a single microarray experiment exceeding $1000 U.S (DFCI) it is in a biologists best interest to identify all potential gene relationships from the data.

The aim of this project is to develop efficient methods for generating gene networks which do not impose a limitation on the number of genes or the consecutive time points that can be analysed. Such a scheme would benefit biologists tremendously.

1.2 Contributions

This work contributes to the field of data mining allowing effective analysis of microarrays to infer gene networks. These contributions are summarised as follows:

1. Three main weaknesses in existing gene network extracting methods have been identified:

    (a) Only a few genes can be analysed at a time

    (b) Scalability is unacceptable

    (c) Only one consecutive time frame can be studied

    These points have been solved in this thesis.

2. A survey of bottom-up and top-down association rule mining algorithms, critiquing their potential for identifying gene networks is provided.

    Top-down approaches are superior to traditional bottom-up methods and other prediction methods as they do not impose a limit on the number of genes studied. However pruning based on support poses other limitations:

    (a) Rules identified describe gene relationships that are mostly well known and are present across the majority of the experiments. A main goal of microarray experiments is to uncover the specific gene relationships of each experimental scenario.

    (b) Many high confidence rules which exhibit low support remain undiscovered, due to support pruning

    (c) Rule set has extremely low recall for gene relationships

3. An investigation of association rule mining algorithms that are designed to identify high confidence rules without support pruning is provided. Previous

methods are bottom-up and as such are not applicable to global analysis of microarrays or other dense datasets.

4. The *Weak Downward Closure* property of confidence is defined, allowing significant improvements on top-down rule mining algorithms (Section 5.2.2).

5. A confidence based top-down algorithm (MAXCONF) for identifying interesting gene relationships on a global genome scale is proposed and implemented. This algorithm efficiently identifies high confidence rules without support pruning achieving significant recall improvements (Section 5.2).

6. Identified the importance of incorporating all time frames when predicting gene networks from sequential microarrays. Previous prediction methods are restricted to analysing only two time frames, losing important information.

7. An efficient top-down *Maximal Sequential Pattern* mining algorithm (SEQRE) is provided which considers all consecutive time frames and can be applied to all genes (2-D global analysis of temporal data). This algorithm will benefit biologists significantly who till now have not being able to analyse temporal microarray data on a 2-D global scale (Section 6.2).

   This is made feasible by:

   (a) incorporating the *Downward Closure* property of support to allow more effective pruning to take place.

8. A systematic framework to validate the rules discovered using two highly regarded biological databases, *BIND* and the *Gene Ontology* has been designed. These databases allow us to evaluate all our results against previous methods with respect to recall and biological significance. (Section 5.4.2)

9. A framework and algorithm is provided that generates *Local Gene Networks* (Section 6.4). These networks effectively describe the identified gene relationships.

   (a) As the number of relationships discovered is extremely large, a Local Gene Network can be constructed around a specific gene of interest.

   (b) A method to visualise the LGN is also provided, clearly showing the various types of relationships.

(c) Can be used for both sequential and non-sequential relationships.

(d) Allows biologists to navigate through the large rule set with ease.

1.3 Thesis Overview

This thesis covers the design and evaluation of two new data mining algorithms capable of extracting gene networks from microarrays that are not discovered with existing methods.

Chapter 2 introduces some key concepts of molecular biology and the current issues concerning biologists, with respect to microarray analysis. The information provided signals the need for effective methods to extract gene networks from microarrays. Finally the microarray datasets used during this study are presented.

Chapter 3 presents a more thorough overview of existing methods to extract gene networks from microarray data. It concludes by formulating the weaknesses of these approaches into a motivated research direction.

Chapter 4 introduces the frequent itemset mining and association rule mining problems.

Chapter 5 presents our top-down approach to mine *Maximal Confidence Rules* from perturbation data. A detailed evaluation of our method is provided, highlighting the advantages of our approach to extracting gene relationships.

Chapter 6 details a top-down sequential pattern mining algorithm to extract gene networks from temporal microarray data. Our method is able to incorporate all genes and consecutive time points into the analysis effectively. Following an evaluation, it concludes by detailing a method to generate and visualise gene networks based on the relationships we discover in this chapter and the previous.

Chapter 7 discusses possible direction for future work. It concludes this thesis by summarising the main contributions of our research and affirming its significance in the fields of data mining and molecular biology.

Chapter 2

Molecular Biology

Molecular Biology is the study of biochemical and molecular interactions within living cells, providing other fields of biology including biomedicine and agriculture with invaluable knowledge to base their research on.

This chapter will introduce some key concepts of molecular biology which build a basis for our research objective. Gene networks are defined and the importance of extracting such networks from microarrays is also emphasised. We describe two well-known microarray datasets which we will use to evaluate our new data mining techniques.

2.1 Yeast

Molecular biology is most effectively studied in small organisms, which rapidly breed. *Sacchrymyces cerevisciae* commonly known as *budding yeast* is one of the simplest organisms with many underlying molecular processes common to humans. As such, yeast is one of the most well-studied organisms, with its entire genome sequenced and many well-regarded microarray datasets publicly available. Furthermore, there is a large repository of comprehensive biochemistry, genetics and molecular cell biology knowledge available which we will take advantage of to evaluate the methods we propose.

2.2 Gene Expression

The simplest self-stabilising unit in an organism is the cell. It must integrate the activity of its components to form a functional entity and respond to multiple signals in a robust manner. Much current research in molecular biology is focused on discovering and understanding the mechanisms by which cells achieve this.

Cells are comprised of DNA, RNA and protein molecules. DNA molecules contain genetic information in the form of genes, which in turn specify the structure and composition of a single protein. *Gene Expression* refers to the process of transcribing a gene's DNA sequence into a *messenger RNA* (mRNA) transcript which serves as a template for protein synthesis. During protein synthesis the resulting mRNA is translated into chains of amino acids, which are then transformed via folding and various chemical reactions into a three dimensional functional protein. All proteins within the cell are the result of the expression of its corresponding gene. An overview of gene expression is illustrated in Figure 2.1.

The functional state of a cell is influenced by the expression profile of its genes, that is, which genes are expressed or not and at what degree. However, it is the resulting proteins which are the active constituents of the cell that essentially control its behaviour and are responsible for almost all cellular duties.

Gene expression is often a highly complex and controlled process. Every cell in an organism contains an identical set of genes, however different types of cells vary in what genes they express and when. The expression level of a single gene is highly dependent on the distribution of the proteins within a cell. The primary form of gene expression *regulation* (*Gene Regulation*) is the control of transcription by either *activation* or *repression*. Generally specific proteins called *Transcription Factors*, induce this control. Gene expression *activation* refers to the initiation of a gene being transcribed. This is usually carried out by activating transcription factors that recruit various other proteins which work together to transcribe the gene. For example in Figure 2.1 the protein labelled X activates the expression of Gene Y, which in turn leads to the formation of protein Z. On the other hand, a transcription factor which interferes with the transcription process of a gene is said to *repress* that genes expression.

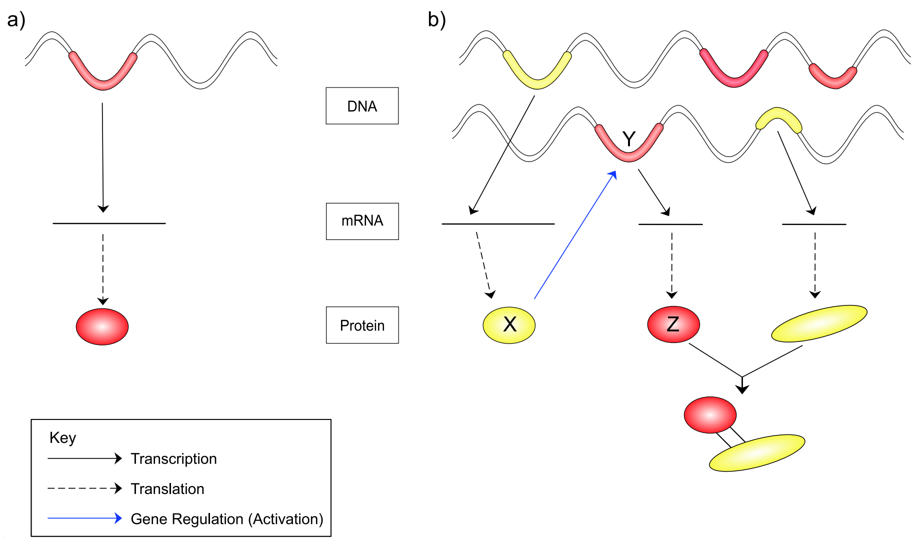


Figure 2.1: (a) Genetic information flow in a cell. Information encoded within the DNA is transcribed into messenger RNA and translated into proteins, the active constituents of a cell. (b) Gene expression complexity is dramatically increased when more than one gene is involved and several proteins are produced. Here the final proteins act together as a single functional unit.

Therefore the regulation of a single gene commonly depends on the presence and absence of various different proteins, and thus requires the prior regulation of the genes encoding these proteins.

2.3 Gene Networks

Molecular biologists are aiming to uncover the functions of unknown genes and their inter-gene dependencies leading up to their expression. Genes work together as a team to perform cellular tasks that no one gene can do alone. To accomplish a single task, many genes may need to be activated or repressed at different times. A *Gene Network* describes the interconnected functional relationships between genes and illustrates the ordered flow of complex events leading up to another gene's expression control or some cellular function. For example part of the network in Figure 2.2 can be interpreted as follows:

FUS3 activates the expression of gene FAR1. FAR1 inhibits CLN1,

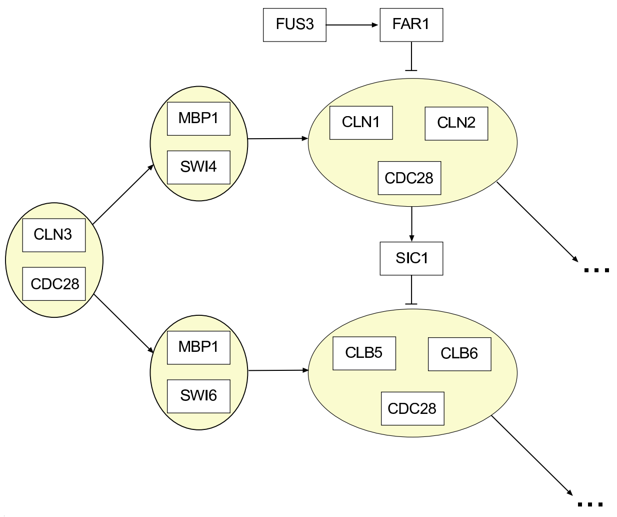


Figure 2.2: Gene Network corresponding to a snippet of the Yeast Cell Cycle. Compiled by Kanehisa and Goto (2000)

> CLN2 and CDC28. In the absence of FAR1, SIC1 will be activated and in-turn inhibit CLB5, CLB6 and CDC28.

The intricate details of gene networks, can assist biomedical research into the underlying cause of disease and to assist in alleviating symptoms and ultimately a cure. This is best explained through the following example:

Diversion from normal physiology is often accompanied by a change in gene expression. Medical researchers are now provided with an opportunity to compare the level of which different genes are expressed between *diseased* and *normal* tissues. For example, more than 50% of all human cancers arise from either a mutation in the p53 gene (rendering the protein product unable to perform its role) or non-expression of the gene in the cancer cells (Lewin, 2000). It is now evident that the protein p53 is vital for controlling cell growth and monitoring a cell's integrity.

The identification of such genetic alterations is attracting the interest of the pharmaceutical industry which see potential drug targets in differentially expressed genes. However knowing which genes vary in expression between healthy and diseased states is only the beginning of developing accurate medication. It is important to understand the processes in which the gene is involved in and needed for. A particular gene may exhibit a dramatic change in expression, but it is possible that this is a result of another gene's or group of genes' expression changes. Therefore, identifying gene networks is a major step in the discovery of gene targeting cures.

2.4 Microarrays

The profile of genes expressed in a cell can provide detailed information about the state of the cell. The presence of a gene's mRNA transcript within a cell indicates that the gene is expressed. Further there is a strong correlation between the expression level of a gene and the amount of mRNA within the cell. The recently developed DNA microarray allows parallel genome-wide gene expression measurements (level of mRNA present) of thousands of genes at a given instant in time, under a given set of conditions for a given tissue/cell. Generation of microarray data introduces a variety of quantitative data analysis issues not encounted in traditional molecular biology or medicine.

The data generated from a series of microarray experiments is commonly in the form of a NxM matrix of expression levels, where the N rows correspond to the genes studied, and the M columns represent the various experimental conditions of the cells under examination. Expression levels generally range between -4 to 4 where positive values correspond to higher expression (when a gene is activated) and negative values correspond to lower expression (when a gene is repressed), compared to a control experiment.

Various experimental designs can be performed using microarrays. These include *Temporal* and *Knockout* experiments.

2.4.1 Knockout Experiments

Gene knockout experiments are based on the intuition that if a gene is no longer able to function normally, it affects the expression of other genes that required its presence. Each individual experiment corresponds to a cell which is genetically altered to prevent the expression of a selected gene. By comparing the expression levels of genes before and after a specific knockout, one should be able to infer what genes the knocked-out gene affects. For example in Figure 2.2, if the gene FAR1 was knocked-out in yeast, we would expect genes CLN1, CLN2 and CDC28 to be expressed.

However this is non-trivial. For example when gene *A* is knocked-out, gene *B* may be expressed, however we can not simply conclude that gene *A* prevents the expression of gene *B* directly. Furthermore, each individual knockout experiment is expensive on its own, as a result it is infeasible for each gene to be studied in this way. Current evaluation methods limit their analysis to the knocked-out genes and those which exhibit obvious changes across the experiments. This potentially leaves many relationships undiscovered. Therefore there is a need for methods that can be applied globally to study all genes on a microarray.

To evaluate our global approaches applicability to knockout microarrays, we apply our methods to the Hughes Compendium of yeast (Hughes et al., 2000). This dataset contains 300 microarrays each corresponding to an individual gene being selectively knocked out. These experiments measured the mRNA levels of 6316 genes of yeast.

2.4.2 Temporal Experiments

It is now becoming popular for microarray data to be in the form of a time series (temporal). The data corresponds to changes in expression levels of genes within a cell overtime typically spanning less than 20 time measurements. These experiments not only allow us to see if gene *A* affects the expression of gene *B*, but if gene *A* must be expressed significantly before gene *B*. For example, in Temporal expression data corresponding to the genes in Figure 2.2, it would be expected that the gene FUS3 is expressed before FAR1, as the presence of the protein FUS3 is required to initiate the expression of FAR1.

We apply our sequential methods to the data of Spellman et al. (1998). This dataset contains 76 gene expression measurements of the mRNA levels of 6177 genes in yeast. These experiments were designed to analyse the changes in gene expression during the cell cycle.

2.4.3 Discretisation

As raw microarray data is continuous, it needs to be discretised before current approaches can be applied successfully. For each dataset we discretise each gene expression value into three categories: down-regulated, up-regulated, neither up or down, by binning an expression value less than -0.2 for the $\log_{10}$ of the fold change, a value greater than 0.2, or a value between -0.2 and 0.2 respectively. The fold change is the ratio of observed expression levels compared to the control. This binning approach is the most commonly used technique (Creighton and Hanash, 2003) when attempting to extract gene networks.

2.4.4 Computational Issues

Microarray experiments are revolutionary as they allow researchers to undertake global gene expression analysis, as opposed to conventional expression methods, which only allow the expression of a few genes to be studied in a single tedious and laborious experiment. Interesting results are beginning to emerge from the use of DNA microarrays to classify subtypes of cancer and to guide treatment decisions (Shipp et al., 2002). Despite this, biologists are experiencing difficultly analysing the copious amounts of expression data generated. Consequently, there is a considerable interest in developing efficient data analysis algorithms to extract complex relationships and to construct gene networks.

Various models, outlined in Chapter 3 have been proposed to solve these issues each differing in their level of abstraction, and thus vary with the information they provide. The algorithms for constructing and generating these models are inefficient, requiring large amounts of memory and processing time. The networks they generate exclude useful information as they are restricted to a small number of genes and consecutive time points. Therefore, there is a need for efficient algorithms that can extract as many gene relationships as possible from microarray data. Moreover, the networks and relationships need to be easily interpreted and more importantly, information regarding a single gene must be easily retrieved from a large gene network.

2.5 Summary

This chapter gave a brief introduction to gene networks and the current issues molecular biologists are facing as their technology advances at a much faster rate then their analysis capabilities. We introduced the microarray technique which contains a large amount of useful information which biologists currently have difficulty in analysing. A description of the various types of microarray datasets we use to evaluate our approaches on was also provided.

Chapter 3

Learning problems associated with Gene Networks

In the previous chapter gene networks and microarrays were introduced. This chapter reviews the existing data mining methods used to extract gene relationships from microarrays to construct gene networks. The individual strengths and weaknesses of the main approaches are highlighted which will provide a strong motivation for the research presented in this thesis.

3.1 Clustering

Clustering is the most commonly used technique for analysing microarray data, grouping either genes with similar expression patterns or experiments with similar gene profiles together. Clustering has been successfully used to classify cancer patients into more specific groups based on the expression profile of known cancer causing genes allowing for more personalised treatment regimes to be used (Slonim et al., 2000). There has been extensive experimentation with various clustering methods including *Hierarchical clustering* (Eisen et al., 1998) and *Self Organising Maps* (Toronen et al., 1999).

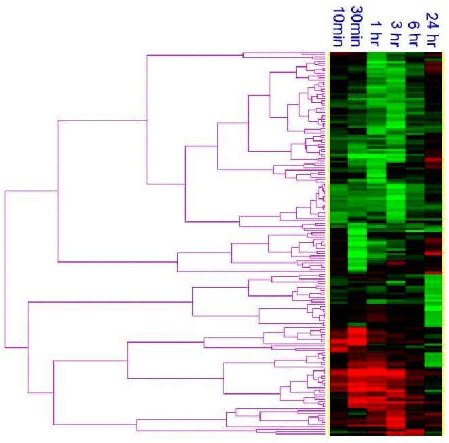


Figure 3.1: Hierarchical tree (LMIL, 2005)

3.1.1 Hierarchical Clustering

Hierarchical clustering is a bottom-up technique that begins by assigning each gene to an individual cluster. It continues by iteratively merging the most similar pairs of clusters until all clusters are combined into one. The result is a hierarchical tree, whose branch lengths represent the degree of similarity between genes or clusters of genes. An example is shown in Figure 3.1. The final tree can be manipulated to identify relationships among genes, for instance, the branches may be sorted to place together gene clusters with similar expression patterns or cut to yield a specific number of clusters for further inspection. Eisen et al. (1998) formed a hierarchical tree of the genes of yeast. Their analysis indicated that genes clustered together not only have similar expression levels but related functions, resulting in the hypothesis that genes with similar expression profiles belong to the same pathway. Similar observations have been made using other clustering methods including *Self-Organising Maps* (Toronen et al., 1999), however hierarchical clustering remains the preferred choice among molecular biologists.

3.1.2 Significance of Clusters

Although clustering has generated groups of functionally related genes based purely on expression profiles, assuming that genes within a clusters belong to the same pathway is incorrect. This assumption is often described by the phrase *Guilt by Association,* (Quackenbush, 2003) which refers to several cases, where clusters can incorrectly reflect the true relationship between genes, as follows:

1. Gene pathways commonly comprise of influencing genes with opposing expression levels. For example, a gene may be unexpressed to allow the expression of another. More precisely, the expression of a gene can repress the expression of another.

2. It is possible for genes to have more than one function (which may still be unknown) and thus appear in more than one pathway.

3. More than one gene pathway can be active at a time. Thus, genes from one pathway may display similar expression profiles to genes in another, and be placed in the same cluster.

Furthermore, generated clusters containing functionally related genes do not show the relationship between the grouped genes. They do not reveal the underlying reason for a gene's observed expression profile, for example, genes A and B in one cluster may belong to the same regulatory pathway, but it is not known if A affects the expression of B, or vice-versa or neither.

3.2 Boolean Networks

Boolean Networks have been used to model the Boolean relationships present in gene networks (Akutsu et al., 2003, 1999, 2000). A Boolean network $G(V,F)$ consists of a set of n nodes ($\mathbf{V}$) corresponding to individual genes each of which has a specific function from the set F of Boolean functions assigned to it. Each gene's expression level is simplified to two states: activated or in-activated.

The Boolean Network model requires microarray data to be in the form of a table, consisting of *Input* and *Output* pairs corresponding to expression patterns of genes

Input Time t-1			**Output Time t**		
X	Y	Z	X’	Y’	Z’
0	1	0	0	0	0
1	1	0	0	0	1
0	1	1	0	1	1
1	1	0	1	1	1
1	1	0	0	1	1

Table 3.1: Boolean expression table

at time $t-1$ and t respectively for various experiments. It may also be applied indirectly to Knockout data which first needs to be converted into an *approximate* time series of *Input* and *Output* pairs (Akutsu et al., 2003). An example dataset is shown in Table 3.1. This dataset is much smaller than one obtained from microarray experiments.

Each Boolean function can receive input from up to K input nodes. That is a single gene can only be influenced by up to K other genes. The binary state of a gene node V_i at time t is determined by the current states of the K input nodes and the corresponding Boolean function $\mathcal{F}$, see Equation 3.1.

$$V_i(t) = \mathcal{F}_i(V_i(t-1), ..., V_k(t-1)) \tag{3.1}$$

For example, if gene Z is always expressed at time t when gene X and Y are expressed at time $t-1$ then an *AND* function with input nodes X and Y will be assigned to node Z. Now suppose that whenever gene W is on at time $t-2$, gene X is not on at time $t-1$, then a *NOT* function with input W will be assigned to node X. This simple Boolean Network is depicted in Figure 3.2.

This model assumes that the expression level of each gene is influenced by K genes. This is a necessary simplification for Boolean Network construction algorithms. This bound will result in important complex gene interactions remaining undiscovered, as it is impossible to know the number of genes which influence a gene prior to construction.

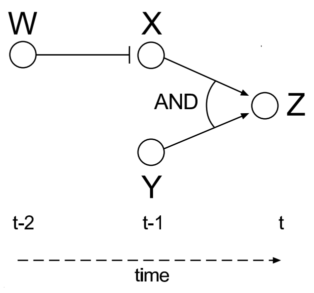


Figure 3.2: Simple boolean network

The algorithm BOOL-1 of Akutsu et al. (1999) infers a Boolean Network from temporal expression patterns. For each node, it exhaustively searches through each set of K input nodes and all possible Boolean functions to identify a function with K inputs, which satisfies all given expression patterns. Akutsu et al. (1999) prove the algorithm is polynomial $O(n^{K+1}m)$, where n = total number of genes, K = number of input genes for each gene, and m is the number of sample experiments each function must satisfy.

As the number of genes studied in a single microarray experiment is commonly in the thousands and the value of K for many genes is known to exceed four, the algorithm is extremely inefficient for inferring large gene networks. In fact experiments from Akutsu et al. (1999, 2000) and Liang et al. (1998) only increased K to at most 4, for this reason. Further Liang et al. (1998) only applied their algorithm to a subset of 50 genes where “the number of configurations becomes too large to compute”.

The REVerse Engineering ALgorithm (REVEAL) (Liang et al., 1998) is similar to the method described by Akutsu et al. (1999). However, it effectively determines the minimum K for each node based on *mutual information* and *entropy* measurements of the input combinations for each output, prior to network construction. The final network corresponds to the minimum network possible and is constructed via an exhaustive search for Boolean functions as in BOOL-1.

Networks generated by algorithms such as BOOL-1 and REVEAL are only capable of modelling how a gene’s expression level at time t (output) is influenced by genes at time $t - 1$ (input). They do not convey how gene profiles at time $t - 1$ are influenced. To do so, an entirely new network needs to be constructed with time $t-1$ as output.

The Boolean network model has many limitations, and as such many models including Noisy (Akutsu et al., 2000), Probabilistic (Shmulevich et al., 2002) and Temporal (Silvescu and Honavar, 2001) Boolean networks have been devised. Each has the ability to increase the accuracy of modelling real gene regulatory networks, and therefore improve the quality of data collected from microarray experiments. However, the time complexity for inferring the networks increases significantly and they are still limited in the number of genes and consecutive time points they can use (Shmulevich et al., 2002; Silvescu and Honavar, 2001).

3.2.1 Noisy Boolean Networks

The standard Boolean model restricts the interactions of genes to that of strict logical rules. The algorithm BOOL-2 (Akutsu et al., 2000) infers a *Noisy Boolean Network* by incorporating boolean functions with inputs which are consistent with at least a predetermined percentage of input/output patterns, whereas BOOL-1 rejects any function which is not consistent with all patterns.

3.2.2 Probabilistic Boolean Networks

Probabilistic Boolean Networks (PBN) remove the assumption that each gene has only one logical rule (Shmulevich et al., 2002). PBNs deal with the uncertainty, both in the data and the model selection by allowing multiple functions predicting each genes expression level. Each node is assigned a set of functions that can predict the expression profiles of the gene, with a certain level of probability.

3.2.3 Temporal Boolean Networks

Each Boolean model discussed so far has assumed that the state of K genes at the previous time phase determines the expression pattern of a gene at a given time.

They are therefore unable to model latency periods between a gene's activation (in-activation) and its observed effect. This is biologically important, as the regulation of many genes requires the expression of more than one gene that can be expressed at different times, in which case the effect is not observed until the later gene is expressed.

Silvescu and Honavar (2001) introduced a Temporal Boolean Network (TBN) model to represent how the expression levels of other genes during preceding time steps can influence the expression of a gene. In a TBN the state of a gene at time t can depend on the states of genes at times $t-1$, $t-2$, ..., $t-(t-n)$, compared to the standard model where they only depend on the states at time $t-1$. TBNs are still unable to reflect complete gene pathways. They only show how input genes regulate genes at time t, not how the input genes are regulated or what genes the output genes regulate. However, they do provide significantly more detail then the standard Boolean network.

3.3 Bayesian Networks

Recently Bayesian networks (Pearl, 1988) have been considered to graphically represent the probabilistic relationships between multiple genes in non-temporal microarray experiments (Friedman et al., 2000, 1999). Bayesian networks are directed acyclic graphs $G(V, E)$ whose vertices V correspond to random variables (genes) and edges E represent the direct dependencies between the variables. The graph G embeds the conditional in-dependencies and dependencies between the variables, and represents the joint probability distribution of all the variables.

Algorithms to identify Bayesian networks aim to maximise statistically motivated scoring functions such as the Bayesian score, which evaluates a given networks ability to portray the data. Searching for the optimal Bayesian network is NP-hard, and therefore heuristic searches and dynamic programming algorithms have been devised to deduce the most likely network.

An attractive property of Bayesian networks is that they are able to model additional attributes, which affect a biological system other than the expression levels. That is, they have the capacity to combine heterogeneous datasets such as epidemiology data. Therefore they have the potential to produce expert systems for diagnostic applications (Baldi, 2002). Unfortunately, this is not appropriate for predicting gene networks from microarray data as additional prior knowledge is likely to lead to a bias towards a specific graph, leaving unknown interactions undiscovered. Furthermore, Bayesian networks are best applied to a small number of variables, thus they suffer from the same input limiting restrictions as Boolean networks.

3.4 Summary

A wide variety of microarray analysis methods have been proposed to infer gene networks. The main techniques, Boolean and Bayesian networks, have the following weaknesses:

1. Only a small percentage of genes can be included in the network discovery process
2. Unable to model genes relationships over more than one consecutive time frames

These input restrictions are required to simplify the computation of the networks. As a result the generated networks will not provide the wealth of knowledge embedded in the microarray experiments. As biologists desire the entire microarray to be analysed, they rarely apply Boolean or Bayesian approaches. They continue to apply clustering methods to unveil the gene relationships. These issues with existing methods signal the need for an efficient approach that can be applied globally to microarray data to extract gene networks.

Chapter 4

Association Rule Mining

Association rules describe relationships between binary attributes extracted from datasets. Association rule mining was originally motivated by a desire to examine the behaviour of customers in terms of the products they purchase together. Mining algorithms have the potential to extract interesting patterns from microarray expression data, which may aid in the identification of gene networks where the expression of a gene can depend on the expression of others:

Gene1 $\Rightarrow$ Gene2 (support 10%, confidence 90%)

The above rule states that when Gene1 is expressed 90% of the time Gene2 is also expressed, and Gene1 and Gene2 are expressed together in 10% of the experiments.

This chapter provides a detailed introduction to the frequent itemset and association rule mining problems. Common definitions and algorithms used to mine rules are also introduced, which lay a foundation for the research presented in this thesis.

4.1 Frequent Itemsets and Association Rules

The problem of mining frequent itemsets and association rules is best described in terms of identifying relationships that describe the purchasing patterns of shoppers. Given all the *items* that a store sells, the set of *transactions* corresponds to each individual group of items that are bought together in a single purchase. From these items and transactions the aim is to find the sets of items that appear frequently together in the transactions known as *frequent itemsets*. From these itemsets, we would like to find *association rules* which describe purchasing patterns in the form of buying $item1 \Rightarrow item2$ and $item3$ are also purchased. When applying association rule mining to microarray data, the set of items is the set of genes studied on the microarray, and each individual experiment is a single transaction. We treat each gene as two separate items - one for the gene being up-regulated and one for the gene being down-regulated therefore within a single transaction only one item for a given gene can be present.

Frequent itemset and association rule mining is formulated as follows:

Let the dataset $D = \{t_1, t_2, ...t_n\}$ be a set of n transactions. Let $\mathcal{I} = \{i_1, i_2, ...i_m\}$ be the set of all possible items (m). Each transaction t, consists of a set of items I from $\mathcal{I}$. The aim is to mine all association rules describing relationships between the items based on the transactions in D.

Association rule mining is commonly split into two tasks:

1. Identify all *frequent itemsets* that satisfy a user defined threshold such as *support*.
2. Generate all *association rules* from the frequent itemsets that satisfy a *confidence* threshold.

Definition 1 (Support) *Let $X \subseteq \mathcal{I}$ be a set of items from D. The support of the itemset X in D is the proportion of transactions that contain X:*

$$support(X) = \frac{\#\, of\, transactions\, containing X}{n} \tag{4.1}$$

Definition 2 (Frequent Itemsets) *The itemset X is frequent if the support of X in D is at least the support threshold (minsup). The set $\mathcal{F}$ of* frequent itemsets *in D is:*

$$\mathcal{F} = \{X \subseteq \mathcal{I} \mid support(X) \geq minsup\} \tag{4.2}$$

Definition 3 (Association Rule) *An* association rule *is an implication between itemsets of the form $I_1 \Rightarrow I_2$ where $I_1, I_2 \subset I$ and $I_1 \cap I_2 = \emptyset$*

Definition 4 (Antecedent of a Rule) *Given an association rule $I_1 \Rightarrow I_2$, the* antecedent *of the rule is the set I_1.*

Definition 5 (Support of a Rule) *The* support *of the rule $I_1 \Rightarrow I_2$ is:*

$$support(I_1 \cup I_2) \tag{4.3}$$

Definition 6 (Confidence of a Rule) *The* confidence *of a rule $I_1 \Rightarrow I_2$ refers to the strength of the association defined as the ratio:*

$$\frac{support(I_1 \cup I_2)}{support(I_1)} \tag{4.4}$$

For example, suppose the itemsets $\{A\}$, $\{B\}$ and $\{A, B\}$ occur is 2, 3 and 2 transactions respectively, and there are a total of 10 transactions. Then the support count of each is $\frac{2}{10}$, $\frac{3}{10}$ and $\frac{2}{10}$ respectively. The confidence of the rule $A \Rightarrow B$ is $\frac{2/10}{3/10}$.

4.2 Itemset Mining - step 1

Algorithms to mine association rules typically search for all frequent itemsets and then from these identify association rules. The problem of mining all frequent itemsets is computationally challenging. The main property causing difficulty is that the search space is exponential with respect to the number of unique items within a dataset. Methods incorporating the *support* threshold into the discovery process often limit this space to a more feasible size. Further the number of itemsets identified can be reduced by neglecting redundant itemsets. Both these aspects will be discussed in further detail in this section.

4.2.1 Search Space

In each dataset there contains exactly $2^{|I|}$ different itemsets which may be checked during the search for frequent itemsets. When the number of items in a dataset is large a naïve approach of generating and counting the supports for all possible itemsets is unreasonable with respect to time and memory requirements.

Definition 7 (Candidate Itemset) *Given an algorithm that computes the frequent itemsets $\mathcal{F}$ from a dataset $\mathcal{D}$, an itemset X is a* candidate itemset *if the algorithm considers whether X is frequent or not.*

For most applications the naïve approach will result in the number of candidate itemsets exceeding the amount of memory available, unless a method for reducing these is considered. Agrawal et al. (1993) presented the *Support Monotonicity* property that is now exploited by many algorithms to reduce the search space.

Definition 8 (Support Monotonicity (Agrawal et al., 1993)) *Given a transaction dataset with items $\mathcal{I}$, let I_1 and I_2 be two itemsets such that $I_1, I_2 \subseteq \mathcal{I}$, then*

$$I_1 \subseteq I_2 \Rightarrow support(I_1) \geq support(I_2) \tag{4.5}$$

Therefore, if a candidate itemset is frequent all its subsets must also be frequent. Conversly if an itemset is infrequent, all its supersets will also be infrequent and thus there is no need for them to be considered as candidate itemsets.

4.2.2 Closed Itemsets

In the generation of frequent itemsets, many itemsets are often redundant in that they provide the same information as another itemset. For instance if the itemsets:

{gene1, gene2}

{gene1, gene2, gene3}

exhibit identical support, the first itemset is considered redundant since the information it provides is contained within the second itemset. In this case the second itemset is a *Closed Itemset*.

Definition 9 (Closed Itemset) *The candidate itemset I_1 is a* closed itemset *if there does not exists an itemset I_2 such that:*

1. $I_1 \not\subset I_2$
2. *support(I_1) $\neq$ support(I_2)*

Definition 10 (Frequent Closed Itemset) *The itemset I_1 is a* frequent closed itemset *if:*

1. *support(I_1) $\geq$ minsup*
2. *I_1 is a closed itemset*

Since there can be millions of frequent itemsets extracted from a single database, having a method to reduce these without a loss of information is important (Pasquier et al., 1999). Therefore, if a mining algorithm can restrict the search to closed itemsets, by only ever considering these, the search space may be reduced significantly.

4.3 Frequent Itemset Algorithms

The frequent itemset mining problem has received a great deal of attention since its introduction by Agrawal et al. (1993). Many new methods have since been published, however for the purpose of background to the methods this thesis introduces, we will discuss the popular *Apriori* (Agrawal et al., 1993) algorithm and the recent *Row Enumeration* approach (Cong et al., 2004) designed specifically to mine frequent closed itemsets. The dataset in Table 4.1(a) will be used for examples in the remainder of this chapter. Table 4.1(b) provides the single itemset counts from the transactions in Table 4.1(a).

Transaction	Items
1	B D E F
2	A C E F
3	A C D E
4	A B C E G
5	A B C D E F
6	B C G
7	D E

(a) Example transactions

Item	Support
A	4
B	4
C	5
D	4
E	6
F	3
G	2

(b) Single itemset supports

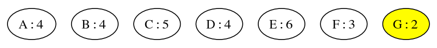
Table 4.1: Example dataset

4.3.1 The Apriori Algorithm

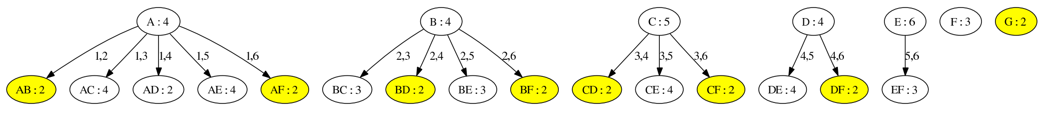
The first phase of the Apriori algorithm (Algorithm alg:apriori) mines all frequent itemsets given a support threshold. The basic approach of this algorithm can be described as searching the *item enumeration* space. Item enumeration refers to the systematic testing of combinations of itemsets, starting from single itemsets and iteratively extending these with one item by combining them with other itemsets. This is referred to as a *bottom-up* technique since it starts with the smallest possible itemsets and builds up to longer ones. An example of Apriori performed on the data in Table 4.1 is shown in Figure 4.1.

The Apriori algorithm begins by identifying the single itemsets which have a support greater than or equal to the support threshold. From these frequent itemsets ($\mathcal{F}_1$), candidate itemsets of length 2 (C_2) are generated (Figure 4.1(b)). The algorithm iteratively continues in a breadth-first manner, generating candidate itemsets C_{k+1} of size $k+1$, from the frequent itemsets $\mathcal{F}_k$. Candidates of size $k+1$ (C_{k+1}) are formed by taking the union $X \cup Y$ of itemsets $X, Y \in \mathcal{F}_k$ if they differ only by one item. For example in Figure 4.1(c) the itemsets *AC* and *AE* each with support count 4, are combined to form the candidate itemset *ACE*. Here the *support monotonicity property* is exploited where any candidate itemset that has a subset which is infrequent is removed from the candidate set, as any extension of an infrequent itemset is guaranteed to be infrequent and thus does not need to be checked. In the case of *ACE*, all of its subsets are supported and thus it becomes a candidate itemset. If for instance the itemset *CE* was not frequent, then *ACE* would be pruned.

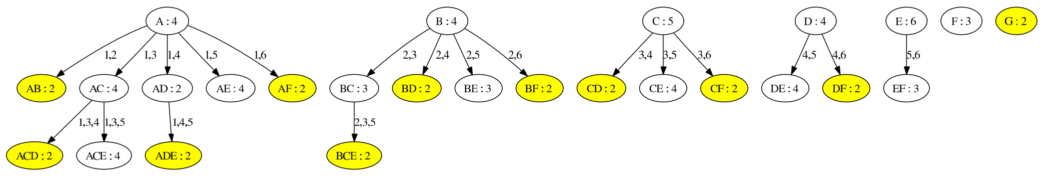
The support of each remaining candidate itemset is then calculated by scanning the transaction database one transaction at a time for each itemset. For each transaction the itemset appears in its support is incremented (Algorithm 1, lines 7-8). All candidates C_k which are frequent are then stored into $\mathcal{F}_k$ (Algorithm 1, line 10). Apriori then continues searching until no further candidates are generated. After all frequent itemsets are identified, association rules are produced which satisfy the confidence threshold from these itemsets.



(a) Iteration K = 1



(b) Iteration K = 2



(c) Iteration K = 3

Figure 4.1: Apriori algorithm example (Min Support = 2)

4.3.1.1 Application to Microarray Data

Creighton and Hanash (2003) applied Apriori rule mining to the expression profiles of 6316 genes (where each gene was represented by two items - up-regulated or down-regulated) corresponding to 300 diverse mutation experiments (transactions) of yeast. Many of the rules generated were consistent with biological knowledge, and other rules revealed numerous unexpected relationships that warranted further biological investigation. Furthermore, genes with previously unknown function were associated with genes of known function, and thus a hypothesis of function was inferred. The associations generated revealed correlations between many genes that were not identified from clustering methods. However their analysis was only applied to 47 rules consisting of at least 11 items, with minimum support 10% and confidence 80% that they discovered (Creighton and Hanash, 2003). Creighton and Hanash (2003) were unable to lower the support threshold below

Algorithm 1 Apriori - itemset mining

```
Input: Database $\mathcal{D}$ and min_support
Assume: Items in transactions and itemsets are sorted.
Output: All frequent itemsets $\mathcal{F}$
 1: $C_1 := \{\{i\}|i \in \mathcal{I}\}$
 2: $k := 1$
 3: while $C_k \neq \emptyset$ do
 4:   // Compute the supports of all candidate itemsets
 5:   for all transactions $t \in \mathcal{D}$ do
 6:     for all candidate itemsets $X \in C_k$ do
 7:       if $X \subseteq t$ then
 8:         X.support := X.support + 1
 9:   // Extract all frequent itemsets
10:   $\mathcal{F}_k := \{X|X.support \geq min\_support\}$
11:   $\mathcal{F}.append(\mathcal{F}_k)$
12:   // Generate new candidate itemsets
13:   $C_{k+1}$ := getCandidates($k + 1$)
14:   k := k + 1
```

10% due to memory requirements.

The generation of significant associations presents the possibility of inferring gene networks from association rules. However as this work demonstrates, Apriori style algorithms, along with support pruning are not appropriate for predicting gene networks.

4.3.2 Row Enumeration

4.3.2.1 Motivation

The Apriori algorithm is very efficient when applied to *sparse* datasets. A dataset is considered sparse when each individual item appears in a small percentage of transactions, where the number of transactions is large (often in the thousands), and each transaction has very few items. In such a dataset the number of frequent itemsets will be low, even when the support threshold is set relatively low.

Despite Creighton and Hanash (2003) showing the usefulness of applying association rules to microarray data, their experiments revealed the computational issues of algorithms based on *item enumeration* like Apriori with such data. The following properties hinder the subsequent iterations of Apriori dramatically with respect to time and space:

1. Dense datasets

   High dimensional microarray datasets impair the tractability of Apriori style algorithms. Compared to traditional datasets mined using Apriori algorithms, microarray datasets contain far fewer transactions (≤ 300 experiments) with each transaction on average consisting of thousands of items. If we consider d to be the maximum number of items in a transaction, then there can be at most 2^d candidates in the item enumeration search space. With d often exceeding 1000, infeasible computational methods would be required to handle these candidates.

2. Long itemsets

   With few transactions and many items, very few items will be deemed infrequent in the first pass (and often many more) unless the support is set to a much higher value which is undesirable. Furthermore, as the expression of many genes are known to be correlated many candidate itemsets of size ≥ 2 are also highly likely to be frequent. As Apriori needs one scan of the database for each item set length to calculate support, long item sets can cause prohibitively long delays.

4.3.2.2 RERII

Both Rioult et al. (2003) and Pan et al. (2003) showed that by searching the *row enumeration* space, the complete set of frequent closed itemsets can be obtained. Compared to *item enumeration* methods like Apriori, *row enumeration* is a top-down approach starting with each transaction being a candidate itemset and iteratively removing items to form smaller candidates of greater support. Cong et al. (2004) presented *RERII*, an efficient row enumeration based algorithm given in Algorithm 2 and 3, best suited to datasets where the number of items greatly outweighs the number of transactions.

RERII constructs a row enumeration tree shown in Figures 4.2, 4.3 and 4.4 in a depth first manner from the dataset in Table 4.1(a), which appear at the end of this chapter.

The algorithm begins by removing all infrequent single items from the transactions and then initialises a set of parent nodes each corresponding to one of the n new transactions with a support count of 1 (Algorithm 2, lines 1-5).

From these parent nodes, candidate subsets (child nodes) of greater support are generated (Algorithm 3, lines 13-19) by taking the intersection between each pair of parent itemsets. If an intersection exists, four scenarios are possible:

Let n_i and n_j be two sibling nodes, where n_i is to the left of n_j.

1. If n_i.item = n_j.item, n_j is integrated into n_i by removing n_j and incrementing the support of n_i and any of n_i's child nodes.

2. If n_i.items $\subset$ n_j.items, the support of n_i and any of its child nodes is incremented.

3. If n_i.items $\supset$ n_j.items, n_j will be pruned and the intersection between n_i and n_j will become a child of n_i with incremented support if it has not been discovered as a frequent itemset in a previous step.

4. If n_i.items $\neq$ n_j.items, the intersection between n_i and n_j will become a child of n_i with incremented support if it has not been discovered as a frequent itemset in a previous step.

After each parent's child nodes are derived the algorithm recurses depth first until no candidate subsets (child nodes) can be formed. At any point if a nodes support exceeds the support threshold it is stored as a frequent closed itemset (Algorithm 2, lines 17-19).

#	Itemset	Support	Apriori	RERII
1	A C	4	Yes	Yes
2	A E	4	Yes	No
3	B C	3	Yes	Yes
4	B E	3	Yes	Yes
5	C E	4	Yes	No
6	D E	4	Yes	Yes
7	E F	3	Yes	Yes
8	A C E	4	Yes	Yes

Table 4.2: Frequent (Closed) Itemsets derived using Apriori and RERII from Table 4.1(a) with support ≥ 3

During *RERII* the support of the parent node under consideration is incremented directly from the nodes it is intersected with (Algorithm 3, lines 2-10), thus eliminating the need to access the database for itemset counting. *RERII* never applies any property of the support monotonicity, although Cong et al. (2004) stated all their pruning is based on support. The pruning in *RERII* is best described as closed pruning, in that it removes all itemsets which are closed by another.

The only point where support is considered is on Line 10 (Algorithm 2), where if it is impossible for a node's support to increase above that of the threshold it is removed. This is the case for node *ABCDEF* in Figure 4.2. As it cannot be intersected with any other nodes, its support can not exceed 1.

Cong et al. (2004) applied *RERII* to real microarray data. However, their analysis only involved performance studies with respect to time and space requirements compared to state of the art Apriori style methods, *CHARM* (Zaki and Hsiao, 2002) and *CLOSET* (Pei et al., 2000). As the support was decreased, *CHARM* failed due to using all available memory and *CLOSET* was found to be too slow, whereas *RERII* performed superior to both.

Table 4.2 highlights the main difference in mining closed patterns with *RERII* compared to mining all frequent itemsets with Apriori. *RERII* does not consider the itemsets *AE* or *CE* as they are already discovered in the closed itemset *ACE*. Note that Apriori style algorithms are not capable of mining closed patterns without having to generate all frequent itemsets individually first.

Algorithm 2 RERII - closed itemset mining - Part 1

```
Input: Min_support, Dataset $\mathcal{D}$
1: for all transactions $t \in \mathcal{D}$ do
2:     Remove all infrequent items from $t$
3:     N := $\emptyset$
4:     n := new Node(items = t.items, support = 1)
5:     N.append(n)
6: FCP := $\emptyset$
7: RERII_depthfirst(N, FCP)
8: Procedure: RERII_depthfirst(N, FCP)
9: for all node $n_i$ in $N$ do
10:    Children := $\emptyset$
11:    if $n_i$ cannot be frequent then
12:        return
13:    for all node $n_j$ in $N$ where $n_j > n_i$ do
14:        $i$ := $n_i$.items $\cap$ $n_j$.items
15:        if $|i| > 1$ then
16:            RERII_pruning($n_i$, $n_j$)
17:    if $n_i$.support $\geq$ min_support then
18:        FCP.append($n_i$)
19:        Disc.append($n_i$.items)
20:    if Children $\neq \emptyset$ then
21:        call RERII_depthfirst(Children, FCP)
```

4.4 Summary

This chapter provided a detailed introduction to the frequent itemset and association rule mining tasks. The relationships association rules describe are shown to be useful for interpreting microarray data. Despite this, the popular Apriori algorithm is shown to be insufficient for mining dense datasets and thus microarrays. This chapter introduced a new family of top-down association rule mining algorithms which were specifically designed to facilitate the mining of dense datasets.

Algorithm 3 RERII - closed itemset mining - Part 2

```
Procedure: RERII_Pruning($n_i$, $n_j$)
if $n_i$.items = $n_j$.items then
    delete $n_j$
    $n_i$.support++
    for all $c$ in Children do
        $c$.support++
if $n_i$.items $\subset$ $n_j$.items then
    $n_i$.support++
    for all $c$ in Children do
        $c$.support++
if $n_i$.items $\supset$ $n_j$.items then
    delete $n_j$
    if $n_i$.items $\cup i$ not discovered before then
        c := new Node(items = $i$.items, support=$n_i$.support + 1)
        Children.append(c)
if $n_i$.items $\neq$ $n_j$.items then
    if $n_i$.items $\cup i$ not discovered before then
        c := new Node(items = $i$.items, support=$n_i$.support + 1)
        Children.append(c)
```

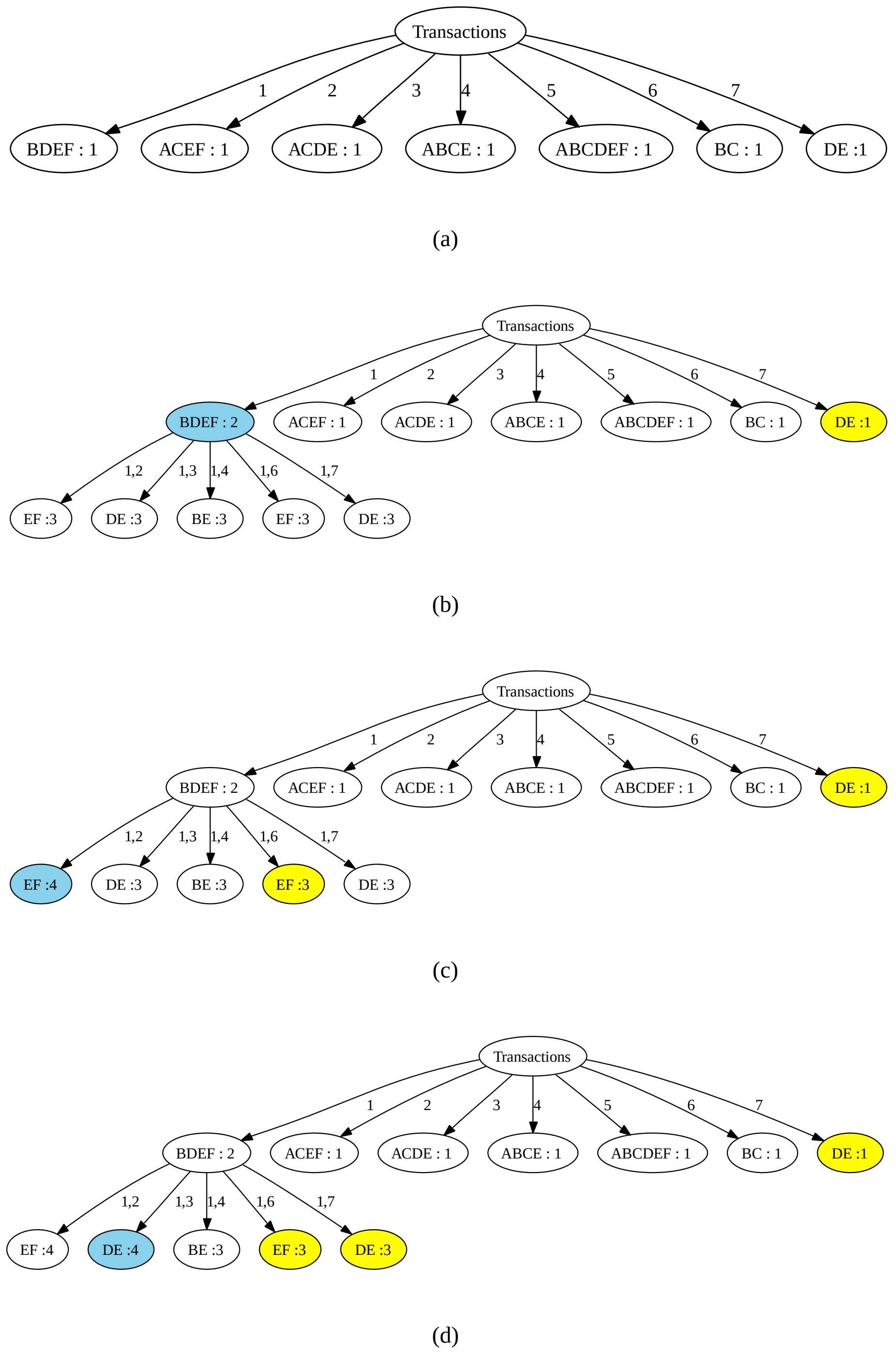
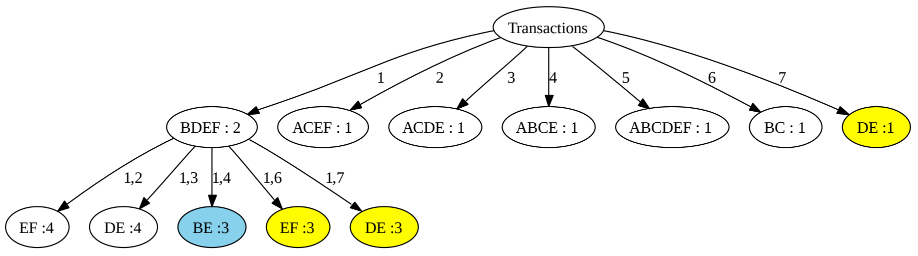
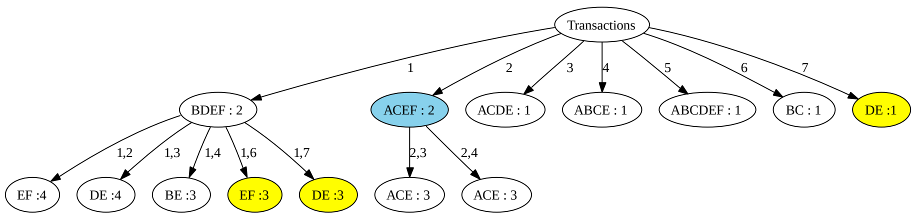


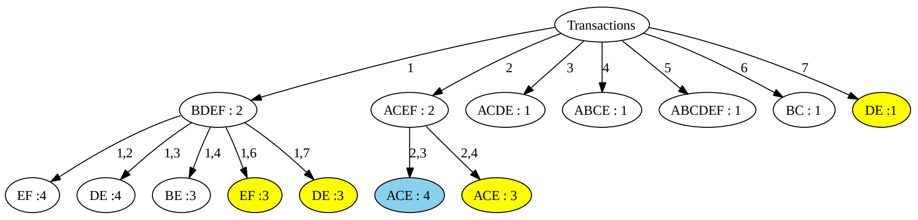
Figure 4.2: RERII algorithm example - Part 1 (Min Support ≥ 3)



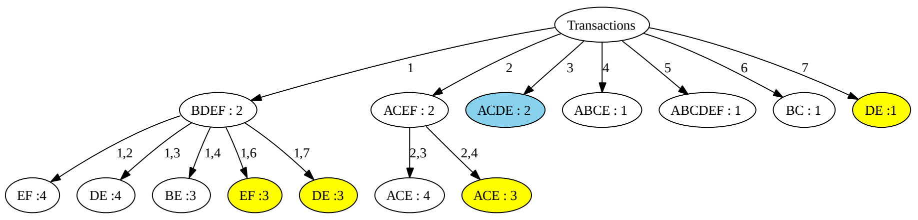
(a)



(b)

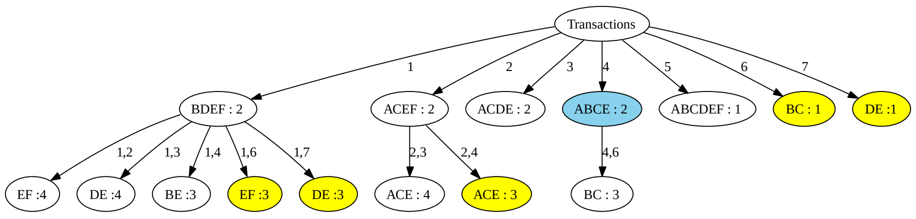


(c)

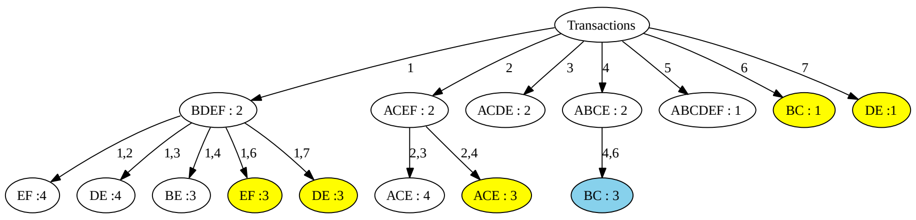


(d)

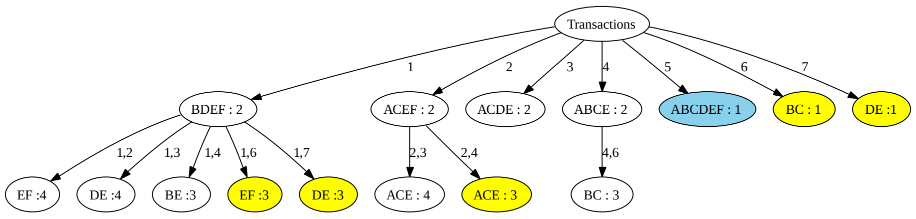
Figure 4.3: RERII algorithm example - Part 2 (Min Support ≥ 3)



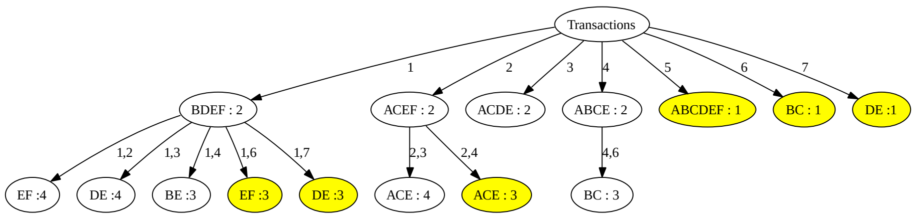
(a)



(b)



(c)



(d)

Figure 4.4: RERII algorithm example - Part 3(Min Support ≥ 3)

Chapter 5

Advances for Gene Networks from Knockout data

We identified two properties of association rule mining algorithms which are not ideal for predicting gene networks.

1. Many rules that a biologist would consider of high interest are pruned, leaving them undiscovered.
2. The enormous number of rules generated hinder a biologists ability to interpret the results.

In this Chapter we address these two short-comings of association rule mining with a strong emphasis on computational effectiveness, proposing a top-down algorithm without support pruning, to mine maximal confidence rules.

5.1 Bottom-up High Confidence Rule Mining

The support based techniques introduced in Chapter 4 deem infrequent itemsets uninteresting, resulting in them being pruned during frequent itemset generation (step 1). Therefore in the following iteration, only a subset of confident rules will be mined. However, it is often the high confidence rules that occur with low frequency which present interesting characteristics within the dataset.

The *Maximal Participation Index* (maxPI) was introduced in Huang et al. (2003) to mine co-location patterns from spatial datasets. It excludes the support threshold from the search, allowing all confident rules to be identified.

Definition 11 (Maximal Participation Index) *Given an itemset I the maximal participation index of I is defined as the* maximal participation ratio *(pr) of all items* $i \in I$.

$$\begin{aligned} maxPI(I) &= max_{i \in I}\{\, pr(I,i)\,\}\ where \\ pr(I,i) &= confidence(i \Rightarrow (I/i)) \\ &= \frac{support(I)}{support(i)} \end{aligned}$$

From Definition 11 it is clear that the MAXPI of an itemset is the maximum confidence a generated rule can have. If the MAXPI of an itemset is below the confidence threshold it cannot generate any confident rules. Unlike support, MAXPI is not monotonic with respect to itemset containment relations. That is:

> Given itemsets I_1 and I_2 such that $I_1 \subset I_2$ we are not guareenteed that MAXPI(I_1) $\geq$ maxPI(I_2)

However, MAXPI does exhibit a *weak monotonic property* (Definition 12). It is possible to applying this weak monotonic property to an Apriori style algorithm to mine confident itemsets. Based on this property, if a k-itemset is MAXPI frequent, then at most one of its subsets with (k-1) items is not confident.

Definition 12 (maxPI weak monotonicity) *Let I_1 be a k-itemset. Then there exists at most one (k-1) subsets I_2 where $I_2 \subset I_1$ such that* MAXPI*(I_2) < maxPI(I_1).*

One drawback of using MAXPI is that no single itemsets can be pruned in the first phase of Apriori as they all have a confidence of 100%. Therefore Apriori-maxPI algorithms must deal with all the singleton candidate itemsets and the $|\mathcal{I}|^2$ 2-itemsets. It is not until 3-itemset candidates are generated that pruning can be applied. Based on support alone, if any (k-1)-itemset of a k-itemset is not frequent, then the k-itemset can not be frequent, and thus can be pruned without a need to do support counting. With respect to MAXPI, a k-itemset is only guaranteed to not be MAXPI frequent (maxPI $\geq$ minimum confidence) if more than k-2 (k-1)-subsets are not. Therefore MAXPI pruning is not as stringent as that using support.

This property works against Apriori, which works efficiently on the assumption that the number of frequent itemsets is low, as stated in Chapter 4. Further with a large number of items in microarray data, Apriori MAXPI approaches suffer from itemset explosion.

Unfortunately there is no property of MAXPI that can be exploited by a top-down approach, without potentially losing confident rules. Motivated by this issue, we have identified a property of confidence that can be exploited by a top-down algorithm. This is described in the following section.

5.2 MAXCONF

The main challenge in devising a top-down algorithm for mining high confident rules is that no support pruning can take place. A naïve approach in a top-down manner would be to grow the entire row-enumeration tree until no itemsets can be generated. This would be equivalent to generating all closed itemsets (including those that are not frequent with respect to support). From these all confident rules may be generated. Concerning microarrays (and other dense datasets), the set of closed itemsets is already extremely large, many of which cannot generate confident rules and as such the naïve approach requires unnecessary expensive computations and memory. We applied this naïve approach, which reported an error after using up all available memory, when only 30% of the transactions had been processed.

In this section we introduce our top-down approach to mining maximal confident rules efficiently. Our algorithm MAXCONF (Algorithm 4) addresses the two main short-comings of association rule mining. MAXCONF exploits two pruning methods each based on confidence allowing us to prune the row-enumeration tree without losing any rules. It is further enhanced to only mine all *maximal confidence rules*. Each of these methods are explained in the following sections.

5.2.1 Level 1 Confidence Pruning

This pruning is based on an observation of the structure of the row-enumeration tree. At any point in the row-enumeration tree we can predict the maximum support and confidence an itemset can exhibit, based on its location within the tree. From this property our first pruning technique is possible, which is detailed in the following definitions.

Definition 13 (Maximum Support) *Given a node N with k child nodes, $N_1, \ldots, N_k$, for any child node N_i the* maximum support *of N_i or any of its potential child nodes is:*

$$maximum\ support \quad = \quad N_i.initial\_support + k - i \tag{5.1}$$

Definition 14 (Minimum Feature) *Given an itemset I, the item $i_1 \in I$ is the* minimum feature *if:*

$$minimum\ feature \quad = \quad support(i_1) \leq support(i_2)\,|\,\forall i_2 \in I \tag{5.2}$$

Definition 15 (Spanning Rule) *Given an itemset I, a rule r spans I if*

$$Antecendent(r)\ \cup\ Consequent(r)\ = I \tag{5.3}$$

$$|Antecendent(r)|\ = 1 \tag{5.4}$$

Definition 16 (Maximum Confidence) *Given a node N, let σ be the maximum support of N and i be the minimum feature of N. The maximum confidence of any spanning rule of N is:*

$$maximum\ confidence \quad = \quad \frac{\sigma}{support(i)} \tag{5.5}$$

$$\tag{5.6}$$

If we know that the maximum confidence of a node's itemset is less than the confidence threshold, it can be pruned, as any further enumeration below the node will only generate less or equally confident child itemsets. This pruning is performed on line 11 of Algorithm 4.

5.2.2 Level 2 Confidence Pruning

We identified the *weak downward closure* property of the confidence measure, which we can exploit during the enumeration tree generation process, to effectively prune nodes which will provide no new information. This pruning is performed on lines 19-21 of Algorithm 4 and is based on the following definitions:

Definition 17 (Max Features) *Given an itemset I, let R_I be the set of all confident rules $\{x \Rightarrow y\}$ where $x \cup y = I$ and $|x| = 1$. The set of* max features *M_I is $Antecedent(R_I)$.*

Lemma 1 (Confidence weak downward closed) *Let M_I be the set of max features derived from I. Then any subset of I which contains an element of M will have a confident rule whose confidence is lower bounded by all rules in R_I.*

Proof 1 *Let $i_1 \in I_1 \cap M_I$. Let r be a rule from I_1 such that Antecedent(r) = i_1 then the rule $i_1 \Rightarrow I_1 \cap$ Consequent(r) is a confident rule because:*

$$\frac{support((I_1 \cap consequent(r)) \cup i_1)}{support(i_1)} > \frac{support(consequent(r) \cup i_1)}{support(i_1)} \tag{5.7}$$

Definition 18 (Sub-rules) *Given an itemset I, let R_I be the set of all rules $\{x \Rightarrow y\}$ where $x \cup y = I$. The set of* sub-rules *SR_I is the set of all rules generated from the itemset I_2 such that:*

$I \subset I_2$

For each $sr \in SR_I$:

$antecedent(sr) \in antecedent(R_I)$

Confidence(sr) $\geq$ Confidence(R)

For example, the rule $A \Rightarrow B$ (confidence 90%) is a sub-rule of $A \Rightarrow B, C, D$ (confidence 80%).

By extension of Lemma 1, if the set of max features M of a node N is not empty, we can prune all child nodes of N whose itemsets are subsets of M, as we are guaranteed that such a child will only produce sub-rules of the rules generated by N.

5.2.3 Maximal Confident Rules

So far our approach has focused on the first issue of association rule mining - the need to identify high confidence rules. We now present another property of confident rules which can be exploited to reduce the number of rules generated, without any information loss - addressing the second issue.

If the set of confident rules can be restricted to that of *Maximal Confident Rules* (Definition 20), the number of rules can be significantly reduced. This approach can only be performed in a top-down algorithm as it exploits the way in which child nodes are constructed.

Definition 19 (Super-rules) *Given an itemset I, let R_I be the set of all rules $\{x \Rightarrow y\}$ where $x \cup y = I$. The set of* super-rules $SupR_I$ *is the set of all rules generated from the itemset I_2 such that:*

$I_2 \subset I_1$

For each $r \in SupR_{I_2}$:

$Antecedent(r) \in Antecedent(R_I)$

Confidence(r) $\leq$ Confidence(R)

$SR_{I_2} = \emptyset$

For example, the rule $A \Rightarrow B, C$ (90% confidence) is a super-rule of $A \Rightarrow B$ (100% confidence). However if the rule $A \Rightarrow B, C, D$ (80% confidence) exists then $A \Rightarrow B, C$ is not a super-rule.

Definition 20 (Maximal Confident Rules) *Let $\mathcal{R}$ be the set of confident rules from a dataset $\mathcal{D}$. The set $\mathcal{MR}$ of* maximal confident rules *is the set of confident rules whose super-rules are not confident.*
For example if the rule $A \Rightarrow$ B,C,D is not confident, but the rule $A \Rightarrow$ B,D is, then the second rule is a maximal confident rule.

During MAXCONF, the first node N down a path which has a max feature set M of cardinality > 1 generates the maximal confident rules R. Let C_M be all child nodes of N with itemsets i such that $i \subset M$. Each child node $c \in C_M$ will generate a

Transaction	Items
1	A B C D E G
2	A C D E G
3	C D E F G H I
4	B C D E G
5	A C E G I
6	A D I
7	D I J
8	A B C D G

Table 5.1: Example transaction set

confident rule (not maximal) which is lower bounded by the confidence of *R* (from Lemma 1). At this point MAXCONF outputs the confident rules of *N* (Algorithm 4, line 18), performs any child pruning (Algorithm 4, lines 19-21), and then continues in a depth first manner. If there remains any child nodes $c \in C_M$ after pruning, all items from *M* are not considered for rule generation from *c* (Algorithm 4, lines 23-24 and 11). Such rules are contained within *R*, and thus can be ignored. Following this procedure, only Maximal Confident Rules will ever be generated.

5.3 MaxConf Example

The complete row enumeration tree for the dataset in Table 5.1 is shown in Figure 5.1. Incorporating the standard closure pruning of RERII, the tree in Figure 5.2 is formed. Suppose confidence = 2/3. Confidence Level 1 pruning will occur on Nodes AI, ADI and ACDG. For example, the support of the itemset ADI needs to be > 2.6 for this node to form any confident rules, as the minimum feature I has a support of 4. Level 2 pruning occurs on node CEG. The parent node of CEG (CDEG) forms 3 confident rules generating the maximal feature set M = {CEG}. Therefore we know CEG will be confident as with any child nodes it may generate, and thus it can be pruned, in which case the node CG will not be created. Now suppose the support threshold = 3 with RERII, no rules from nodes CEGI, DI, BCDEG, DIJ, ACDEG, BCDEG or ABCDG would be generated based on

MAXCONF Rules	Confi dence	Support	RERII Found
C ⇒ DEG	4/6	4	Y
E ⇒ CDG	4/5	4	Y
G ⇒ CDE	4/4	4	Y
A ⇒ CG	4/5	4	Y
C ⇒ AG	4/6	4	Y
G ⇒ AC	4/6	4	Y
A ⇒ D	4/5	4	Y
B ⇒ CDEG	2/3	2	N
B ⇒ CDG	3/3	3	N
I ⇒ D	3/4	3	N
J ⇒ DI	1/1	1	N
F ⇒ CDEGHI	1/1	1	N
H ⇒ CDEFGI	1/1	1	N

Table 5.2: Rules identified by MAXCONF and RERII

support alone. Furthermore the single itemsets B, F, H and I would be immediately pruned in the first pass (unsupported). Figure 5.3 shows the MAXCONF tree with closure and confidence pruning and Table 5.2 shows the various rules identified by MAXCONF that can and cannot be identified using RERII with a support of 3. Note due to confiden ce pruning, MAXCONF will not identify the rule $C \Rightarrow EG$. However this information is contained within the first rule in Table 5.2.

5.4 MAXCONF Evaluation

In this section we concentrate on the general effectiveness of our algorithm compared to previous methods. We demonstrate the importance of pruning without support with respect to performance and the biological significance of the rules generated.

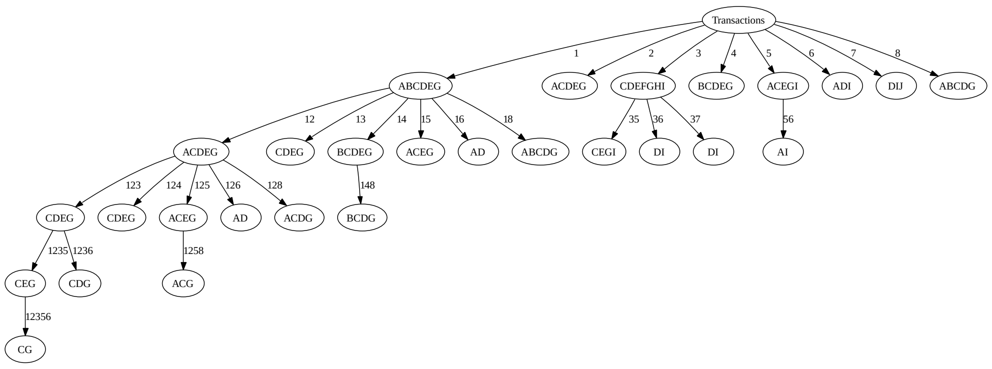


Figure 5.1: MAXCONF tree - no pruning

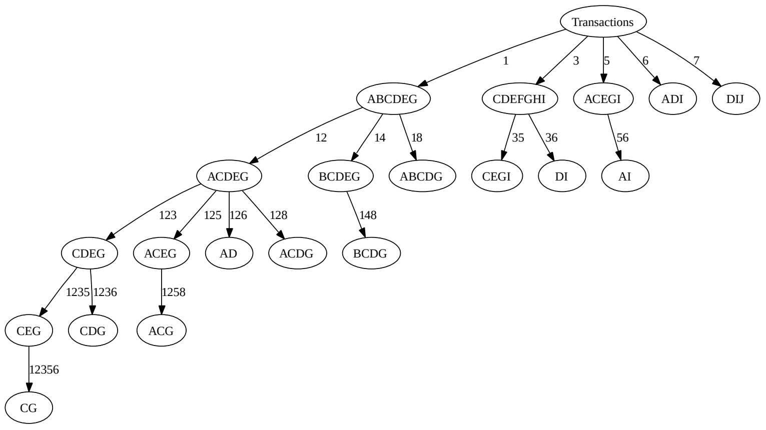


Figure 5.2: MAXCONF tree - closure pruning

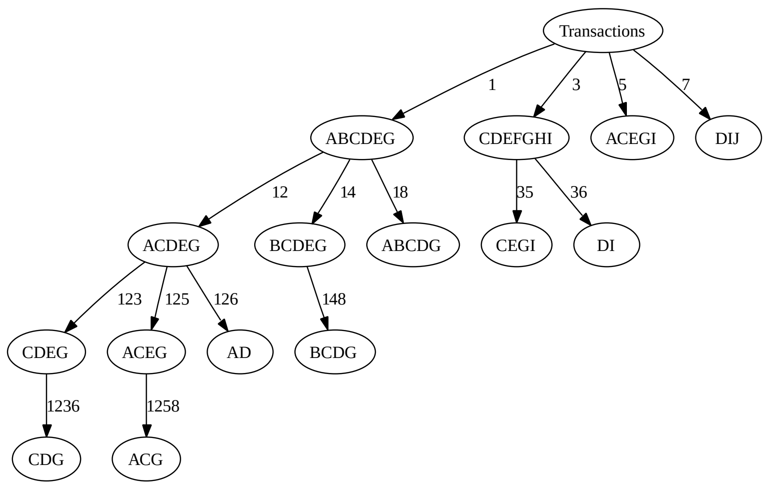


Figure 5.3: MAXCONF tree - confidence pruning

5.4.1 Computational Effectiveness

The main down fall of RERII is its inability to extract many possible association rules that meet the confidence threshold due to its support pruning. Indeed with the Hughes Compendium (Hughes et al., 2000) where 8678 items are considered within 300 transactions, 96.5% of the single items are pruned in the first stage with a high support of 10%, leaving only 301 items to be considered to form frequent itemsets and then confident rules. Without any support cut-off necessary MAX-CONF mines rules considering all 8678 items, and as such is capable of detecting many more rules with high confidence. Furthermore we compared the effectiveness of mining only maximal confident rules to mining all high confident rules. As expected with a lower confidence threshold, fewer rules are generated as more maximal rules are identified. These results are summarised in Table 5.3.

A more detailed comparison of support and confidence pruning is shown Figure 5.5. This graph clearly highlights the drastic effects of support pruning on rule generation. When the support of RERII is lowered to zero (in an attempt to find all confident rules), no rules were ever generated as the program required too much memory.

The difference in the number of rules generated with and without the maximal rule restriction is only moderate (Table 5.3), with only a 11% reduction with 85% confidence. The amount of reduction (as with any pruning) is bounded by the characteristics of the dataset.

Table 5.4 further accentuates the significant improvement of MAXCONF over RERII by identifying high confidence rules with a much larger support range, with a very low minimum.

Confi dence (%)	# Rules		
	RERII[a]	MAXCONF[b]	MAXCONF[c]
80	8083	21448	19090
85	3161	13181	12424
90	927	8445	8296
95	277	7229	7214
100	65	7067	7067

[a]Support 5%
[b]No maximal restriction
[c]Maximal restriction

Table 5.3: Effect of confidence pruning for rule extraction

Confi dence (%)	Support Range		
	RERII[a]	MAXCONF[b]	MAXCONF[c]
80	5 - 30.4	0.3 - 30.4	0.3 - 30.4
85	5 - 30	0.3 - 30	0.3 - 30
90	5 - 25	0.3 - 25	0.3 - 25
95	5 - 25	0.3 - 25	0.3 - 25
100	5 - 17	0.3 - 17	0.3 - 17

[a]Support 5%
[b]No maximal restriction
[c]Maximal restriction

Table 5.4: Range of rule supports

Figure 5.4 shows the scalability ofRERII and MAXCONF algorithms with respect to confidence, support and maximal rule mining. Intuitively with support pruning the higher the support is set, more pruning is possible and thus the run time is

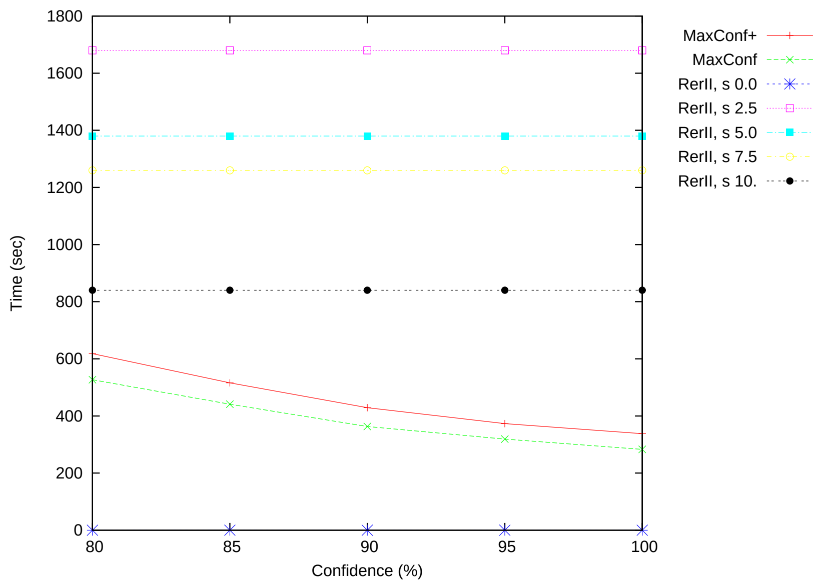


Figure 5.4: Scalability of confidence and support pruning methods

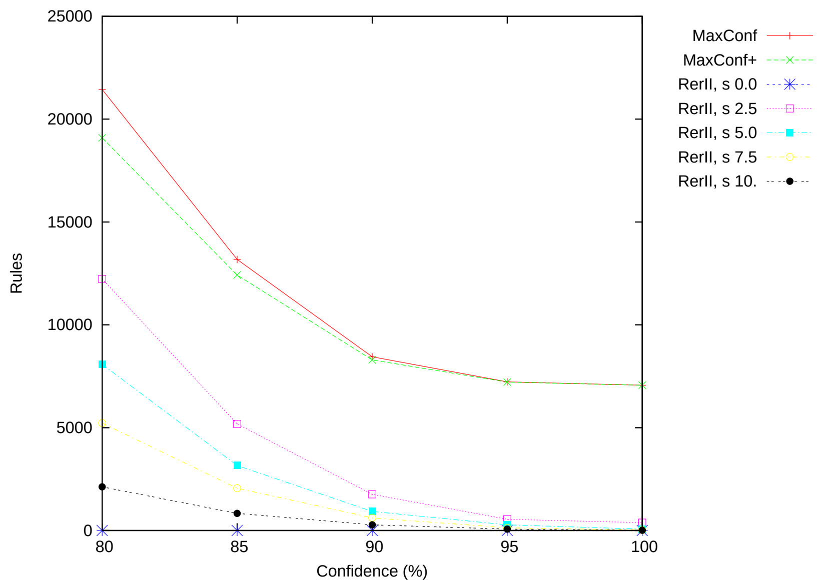


Figure 5.5: Rules discovered with confidence and support pruning methods

decreased.

Surprisingly there was a significant improvement in run time with MAXCONF both with and without the maximal restriction over RERII. This was unexpected as previous approaches to pruning with confidence such as the MAXPI algorithm are often less efficient than their support based alternative.

The improvement is likely to be due to the nature of Level 2 pruning. In RERII, to generate rules satisfying confidence the complete path from the top nodes to the bottom need to be constructed, regardless of whether a nodes itemset is supported or not. However in MAXCONF, when a node satisfies Level 2 pruning, all child nodes are pruned and thus it is impossible for the tree to extend any further. This is indeed significantly advantageous in this case.

As expected, imposing the maximal restriction on MAXCONF slightly increases run time, due to the extra checking required. However this approach is still more efficient than RERII.

5.4.2 Biological Rule Evaluation

In this section we focus on estimating the biological relevance of the rules we identify. Firstly we concentrate on how effective our approach is in detecting known direct biological interactions.

Secondly we show that many of our rules contain relationships between the genes they contain, which are not direct interactions. In each analysis we give a few examples of rules we identify, and how they relate biologically. Finally we consider the *iron uptake pathway* in yeast, presenting some of the rules identified by MAXCONF that correctly describe gene relationships in this system. For consistency, all further analysis of RERII involves the rules extracted with a minimum support of 5%.

5.4.3 BIND

The Biomolecular Interaction Network Database (BIND) (C Alfarano et.al., 2005) is an online database that archives pairwise information about direct[1] interactions which can occur between two biological entities, including proteins, RNA, DNA, and genes. All interactions documented are experimentally determined using traditional wet-lab experiments, with the minimum amount of information required to define an interaction being a PubMed publication.

5.4.3.1 Direct Interactions

To verify our method's ability to extract meaningful biological results and its applicability to gene networks, we determined how many rules exhibit direct interactions between at least two of their items i.e. *precision* (Definition 21). The intuition behind this analysis is based on the observation that it is highly probable that for a direct interaction between two or more gene products (proteins) to occur, the expression of the genes are correlated, and hence will be present together in at least one rule.

Further we analysed the effectiveness of our approach to identify all possible interactions from the dataset, i.e. *recall* (Definition 22). In total, among all the genes in the microarray data, there are 1354 possible unique direct interactions that can be extracted given the experimental conditions of the microarrays we analyse. These results are summarised in Table 5.5 and clearly show the effectiveness of MAX-CONF over support based mining methods. The extremely high recall (94%) is superior compared to that obtained using RERII (0.15%).

Definition 21 (Precision) *Let $\mathcal{R}$ be the set of rules identified by a mining algorithm. Let $\mathcal{B}$ be the set of pairwise direct interactions in the microarray dataset, in the form of rules. The percentage of rules which contain a direct interaction is:*

$$\frac{\#\ \textit{rules in}\ \mathcal{R} \cap \mathcal{B}}{\#\ \textit{rules in}\ \mathcal{R}} \tag{5.8}$$

[1]a direct interaction refers to when two biological entities must physically bind together to allow some function

Definition 22 (Recall) *Let $\mathcal{R}$ be the set of rules identified by a mining algorithm. Let $\mathcal{B}$ be the set of pairwise direct interactions in the microarray dataset, in the form of rules. The* recall *of direct interactions in $\mathcal{R}$ is:*

$$Recall = \frac{\#\ rules\ in\ \mathcal{R} \cap \mathcal{B}}{\#\ rules\ in\ \mathcal{B}} \tag{5.9}$$

Confi dence (%)	Precision (%)		Recall(%)	
	RERII[a]	MAXCONF	RERII[b]	MAXCONF
80	29.8	80.1	0.15	94.0
85	37.9	82.5	0.15	94.0
90	23.3	84.1	0.15	94.0
95	26.8	84.1	0.15	94.0
100	18.2	84.1	0.15	94.0

[a]Support 5%
[b]Support 5%

Table 5.5: Direct interactions identified in rules

The low recall of RERII was surprising considering the percentage of rules found that contained at least one direct interaction (37.9% with 85% confidence). After further inspection of these rules it was clear that many of the interactions were not detected as 96.5% of the genes were immediately pruned (not satisfying support) before RERII began. Furthermore, 99.96% of the rules from RERII containing direct interactions, included the genes SNO1 and SNZ1.

Examples of rules displaying direct interactions are shown in Table 5.6. Both Rules 1 and 3 in Table 5.6 would not be identified unless the support threshold for RERII was decreased significantly (if possible with respect to memory requirements). Rule 3 with 100% confidence correctly describes the relationships between the genes (CSE1 binds PCL5, which in-turn PCL5 is able to bind CRM1). Rule 2 with its high support, is the most common rule published to validate previous approaches (Creighton and Hanash, 2003). The co-expression of CTF13 has been hypothesised to be caused by the close proximity of CTF13 to SNO1 and SNZ1 (Creighton and Hanash, 2003).

#	Association Rule	Supp (%)	Conf (%)	BIND Interaction
1	FMP17 ⇒ ERG28 ERG25	0.60	100	ERG28:ERG25
2	CTF13 ⇒ SNO1 SNZ1	21.0	80.8	SNO1:SNZ1
3	CSE1 ⇒ CRM1 PCL5	0.33	100	PCL5:CSE1 PCL5:CRM1

Table 5.6: Association rules exhibiting direct interactions in yeast

Association Rule		Supp (%)	Conf (%)
1	EUG1 ⇒ BNA2, GSC2, PDH1, TFS1, THI5, THI11, THI13, YGR043C, YML131W	1.30	100
2	SIL1 ⇒ AFR1, GSC2, YPS1, YOR289W	2.67	100

Table 5.7: Association rules exhibiting indirect interactions in yeast

5.4.3.2 Indirect Interactions

Many extracted association rules contain genes which interact indirectly via other genes and their products. Table 5.7 shows two of these rules.

In Rule 1 of Table 5.7 the proteins encoded by genes BNA2, EUG1, PDH1, THI5, THI13 and YML131W all bind the protein PRP20. THI11 binds directly to SNZ2 which binds PRP20. The remaining genes GSC2, TFS1 and YGR043C each bind directly to NUP100. The gene NUP100 was not included in the microarray data. However the genes SNZ2 and PRP20 were. Further, each gene in Rule 1 is involved in cellular metabolism, with the gene BNA2, THI5, THI11 and THI13 being specifically involved in *water soluble vitamin biosynthesis* (GO, 2004).

In Rule 2 genes LSM8 and LSM2 connect genes YPS1 and SIL1 indirectly. The protein products of LSM8 and LSM2 bind directly to each other. LSM8 then directly interacts with SIL1 and LSM2 directly interacts with YPS1. Therefore, Rule 2 has successfully identified the indirect interaction between genes SIL1 and YPS1 via LSM8 and LSM2 respectively. Further analysis of Rule 2 also shows that the remaining proteins encoded by the genes AFR1, GSC2 and YOR289W all directly bind the protein product of STE12.

5.4.4 GOstat

The Gene Ontology (GO, 2004) is an international standard to annotate genes organised by their molecular function, biological process and cellular components. For every gene in the GO database there is a link to its associated gene ontologies that define it's function. The GO has a hierarchical structure starting with top level ontologies to specific descriptions.

GOstat (Beissbarth and Speed, 2004) is a query engine wrapper of the GO database where by for a group of genes, GO annotations that are statistically over-represented within the group can be obtained. This tool provides a useful method for analysing the gene groups we identify.

The number of itemsets that formed rules which contained at least two genes that were considered to be statistically over-represented by a GO are shown in Table 5.8. As expected, MAXCONF was able to identify many more relationships, some of which are shown in Table 5.10. For example two genes in Rule 2 of Table 5.10 belong to the same ontology class (nuclear acid metabolism) which has a depth of 5 within the entire GO.

Confi dence (%)	Absolute #Itemsets with GO		Itemsets with GO Cluster (%)	
	RERII [2]	MAXCONF [3]	RERII [4]	MAXCONF [5]
80	323	899	63.5	74.9
85	187	773	65.6	78.3
90	69	693	59.4	82.7
95	34	690	60.7	82.9
100	7	690	63.6	82.9

Table 5.8: GO clusters identified in rules

#	Rule	Supp (%)	Conf (%)
1	FRE6 ⇒ SIT1, ARN1, ARN2, ENB1, FIT2, FIT3	4.33	100
2	AKR1 ⇒ CCC2, SIT1, FTR1, ARN1, ARN2, FET3, ENB1, FIT2, FIT3	3.33	90
3	$\overline{MAC}1 \Rightarrow \overline{FRE}7$	0.33	100

Table 5.9: Association rules related to the Iron Uptake Pathway in yeast

5.4.4.1 Iron Uptake Pathway

S.cerevisiae has two different mechanisms to take up iron from the external environment for it to use in other processes, which combined form the *iron uptake pathway*. A small sample of the rules identified by our system applicable to this pathway are shown in Table 5.9

One system of the iron uptake pathway in yeast depends on a group of proteins, specifically a family of high-affinity transporters encoded by the genes ARN1, ARN2, SIT1 and ENB1. Therefore for this uptake sub-system to function each of those genes need to be co-expressed.

Another sub-system of iron uptake requires some if not all the proteins FRE1-6, FET3, FIT2-3 and FTR1.MAXCONF was able to detect such biological significant patterns two of which are shown in Table 5.9 (Rules 1 and 2). These two rules would not have been detected using the bottom-up Apriori approach to frequent itemset mining as they would have needed to be pruned with respect to support to reduce the search space.

Rule 3 in Table 5.9 is one of the many relationships which indicates the applicability of our approach to perturbation microarray experiments. The gene MAC1 was one of the genes chosen to be perturbed in the Hughes Compendium (Hughes et al., 2000). While extracting many other gene relationships, we were also able to detect relationships that boolean networks attempt to identify. Indeed Rule 3 correctly describes the relationship between the genes MAC1 and FRE7, that is MAC1 is a transcription factor which activates FRE7. Therefore FRE7 will only ever be expressed if MAC1 is prior.

5.5 Summary

This chapter proposed a comprehensive algorithm to mine high confidence rules from microarray data, without the need for support pruning. MAXCONF performs two forms of confidence pruning to successfully reduce the search space. We introduce two biological databases which are used to evaluate our approach with respect to identifying gene relationships. The performance of MAXCONF is compared to RERII and is shown to extract significantly more interesting gene relationships, in significantly less time. A significant increase in recall is achieved using MAXCONF (94%) compared to RERII (0.15%).

Algorithm 4 MAXCONF - highly confident itemset mining

```
for all transactions $t \in \mathcal{D}$ do
    // No single item set pruning
    N := $\emptyset$
    n := new Node(items = t.items, support = 1)
    N.append(n)
MR := $\emptyset$ // set of maximal rules
MAXCONF_depthfirst(N)
Procedure: MAXCONF_depthfirst(N)
for all node $n_i \in N$ do
    Children := $\emptyset$
    if $n_i$ cannot be confident then
        continue // Level 1 pruning
    Determine support of $n_i$, populate Children, and prune based on closure only
    as in RERII.
    M:= getMaxFeatures($n_i$) // Line 27
    if M $\neq \emptyset$ then
        for all $m \in M$ do
            if $m \notin n_i$.maxFeatures then
                CR.append($m \Rightarrow \{n_i.items \setminus m\}$)
        for all child $c \in Children$ do
            if $c \subset$ M then
                delete $c$ // Level 2 pruning
            else
                c.maxFeatures.insert($n_i$.maxFeatures)
                c.maxFeatures.insert($c \cup$ M)
    if Childen $\neq \emptyset$ then
        MAXCONF(Children)
Procedure: getMaxFeatures(n)
M := $\emptyset$ // set of maximal features
for all items $i \in n.items$ do
    if support($n$) / support($i$) $\geq$ min_confidence then
        M.insert($i$)
return M
```

#	Rule	Supp (%)	Conf (%)	GO Cluster			
				Genes	GO	Depth	P-value
1	MEP2 $\Rightarrow$ GLK1 GLC3 DMC1 HSP12 PRY1 NCA3 TFS1 MSC1 PGM2 YGP1	1	100	DMC1 MSC1	meotic recombination	9	0.0475
				GLK1 GLC3 PGM2	carbohydrate metabolism	5	0.0475
				HSP12 YGP1	cell communication	3	0.141
				HSP12 MEP2	plasma membrane	4	0.172
2	$\overline{ESC8} \Rightarrow \overline{IMD1}\,\overline{IMD2}$	1.3	100	ESC8 IMD2	nuclear acid metabolism	5	0.02
				IMD1	unknown		
3	$\overline{POP1} \Rightarrow \overline{RRP7}$	2	85.7	POP1 RRP7	rRNA processing	8	0.0152
4	EUG1 $\Rightarrow$ BNA2, GSC2, PDH1, TFS1, THI5, THI11, THI13, YGR043C, YML131W	1.30	100	THI5 BNA2 THI13 THI11	water soluble vitamin metabolism	7	8.93e-06
				BNA2 GCS2 THI5 THI13 THI11	cellular biosynthesis	5	0.067
				THI5 EUG1 BNA2 TFS1 GSC2 THI13 THI11 PDH1	cellular metabolism	4	0.2

Table 5.10: Example rules containing GO clusters

Chapter 6

Advances for Gene Networks from Temporal data

The rapid growth of temporal data being generated, has lead to increased efforts in methods for discovering the hidden sequential patterns within the data. Existing approaches are targeted to the general forms of sequential data, where the number of items at a single time point (itemset) is comparatively smaller than the length of the sequence and the number of transactions. As with standard frequent itemset mining, algorithms designed for such data are not suitable for temporal microarray analysis.

Temporal microarrays contain many relationships that are extremely useful in elucidating gene networks. Most algorithms for generating gene networks were originally designed for this form of data. However these algorithms are bound by two limitations:

1. The number of genes that can be analysed is restricted
2. The time space they are applied to is restricted

For example the Boolean network algorithms only considers how the behaviour of genes at one time point relate to the expression of genes at another, i.e. only the changes in gene expression between two consecutive time measurements.

In this chapter we introduce and evaluate our algorithm SEQRE which mines sequential patterns using a top-down approach while addressing these two issues. We conclude this chapter by introduceing

6.1 Sequential Pattern Mining

Sequential pattern mining algorithms are used to mine data that has some sequential nature. In the context of analysing customer purchasing behaviour, each sequential transaction is an ordered set of individual transactions by time, for example:

> 5% of customers bought “The Fellowship of the Ring” and “The Two Towers” in one transaction followed by “The Return of the King” in a later transaction.

This problem was first introduced by Agrawal and Srikant (1995), and can be formalised as follows:

Each sequential pattern *s*, is a list of transactions, where each transaction is a set of items present at a given time. A sequence with *k* transactions is called a *k*-sequence. The i^{th} transaction in a sequence is represented by s_i.

Definition 23 (Subsequence) *A sequence* $a =< a_1, a_2..., a_n >$ *is a* subsequence *of* $b =< b_1, b_2..., b_m >$ *if there exists integers* $1 \leq i_1 < i_2 < ... < i_n \leq m$ *such that* $a_1 \subseteq b_{i1}, a_2 \subseteq b_{i2}, ... \; a_n \subseteq b_{in}$

A subsequence s *of* S *is denoted by* $s \subseteq S$.

Definition 24 (Super-sequence) *Given a sequence* s *and* S *such that* $s \subseteq S$ *then* S *is a* super-sequence *of* s.

Definition 25 (Closed Sequence) *A sequence* S_1 *is a* closed sequence *if there exists no sequence* S_2 *such that* $S_1 \subset S_2$ *and support(*S_1*)* $\neq$ *support(*S_2*)*

The *gap constraint* is often applied during the discovery process, to impose a limit on the maximal distance between two consecutive transactions in a sequence. Using a gap constraint, it is possible to specify how much of an impact an event has on closer events in time than on distant ones (Antunes and Oliveria, 2004). In using the gap constraint the notion of a subsequence needs to be adapted to a *δ-distance subsequence*.

Definition 26 (δ-distance subsequence) *A sequence $a =< a_1, a_2..., a_n >$ is a δ-*distance subsequence *of $b =< b_1, b_2..., b_m >$ if there exists integers $1 \leq i_1 < i_2 < ... < i_n \leq m$ such that $a_1 \subseteq b_{i1}$, $a_2 \subseteq b_{i2}$, ... $a_n \subseteq b_{in}$ and $i_k - i_{k-1} \leq \delta$*

A δ-subsequence s of S is denoted by $s \subseteq_\delta S$. When $\delta = 1$, no gaps occur between consecutive items.

Example 1 *Given the sequences, where the itemsets are ordered with respect to time, and an itemset containing more than one item at a given time is inclosed with brackets,*

1. $a(bc)dc$
2. $a(bc)$
3. $a(abc)(ac)d(cf)$

Sequence 2 is a subsequence of sequence 1
Sequence 1 is a 2-distance subsequence of sequence 3.

Definition 27 (Frequent Sequential Pattern) *A sequential pattern S is frequent if the support of S is $\geq$ the minimum support.*

A confidence threshold is not applicable to sequential pattern mining.

6.1.1 Search Space

The frequent itemset mining problem introduced in Chapter 4 is a particular case of sequential pattern mining. Mining 1-sequential patterns is simply the search for frequent itemsets. Considering this, sequential pattern mining is much more challenging than frequent itemset mining. It not only involves the discovery of frequent itemsets, but also the arrangement of those itemsets into sequences and the discovery of which of those are frequent (Antunes and Oliveria, 2004). Therefore the search space significantly increases.

Following Antunes and Oliveria (2004), given a database of sequences with at most

m transactions and n different items, where each transaction has only one item, there are n^m possible different sequences with m transactions and N different arbitrary length sequence (Equation 6.1).

$$N = \sum_{k=1}^{m} n^k = \frac{n^{m+1} - n}{n - 1} \tag{6.1}$$

Now considering that each transaction can contain an arbitrary number of items, there would be S_m possible frequent sequences, as in Equation 6.2, and there would exist $\Theta(2^{nm})$ sequences in general (Equation 6.3).

$$S_m = (2^n - 1)^m \tag{6.2}$$

$$\sum_{k=1}^{m} (2^n - 1)^k = \frac{(2^n - 1)^{m+1} - 2^n - 1}{2^n - 2} = \Theta(2^{nm}) \tag{6.3}$$

The support monotonic property of frequent pattern mining is easily extended to mining sequential patterns, and thus many algorithms based on support pruning of patterns are available. However the difficulties of Apriori style algorithms exhibit when mining frequent itemsets from microarray data, extends also to sequential pattern mining of temporal microarrays.

6.2 SeqRE

A top-down approach to sequential pattern mining is outlined in this section. Our algorithm SeqRE (Algorithms 5 and 6) successfully overcomes the limitations imposed by other gene network algorithms such as Boolean networks and Apriori style sequential pattern algorithms.

The algorithm RERII (Cong et al., 2004) cannot simply be applied to mine sequential patterns. SeqRE is possible with a modification of the row-enumeration tree introduced in Cong et al. (2004), along with an extra support pruning method which can only be applied to top-down sequential pattern mining. These properties of SeqRE will be discussed in the following sections.

6.2.1 SEQRE Algorithm

SEQRE begins like RERII, where all single items that are not supported are removed from each transaction, followed by creating the first set of parent nodes corresponding to the new transactions each with a support count of 1. Following the depth first approach of RERII, a parent's child nodes are formed by taking the intersection between the parent and each of its sibling nodes. It is here that RERII is no-longer applicable to sequential transactions. Two sibling sequential patterns can have more than one pattern as an intersection. For example Figure 6.1 shows the process of RERII with four sequential transactions in Table 6.1 as input. For an interesting comparison, a snippet of a sequential-item-enumeration tree generated by a state-of-the-art Apriori style sequential algorithm (*SPAM*) (Ayres et al., 2002) applied to the same dataset is shown Figure 6.2. From this it is clear that the sequential-item-enumeration tree will become significantly larger than the one generated by SEQRE when all items are considered.

Transaction	Sequence
1	A, (ABC), A, B, D, E
2	A, (ABC), D
3	A, (BE), D, E
4	A, B

Table 6.1: Example sequential transaction set

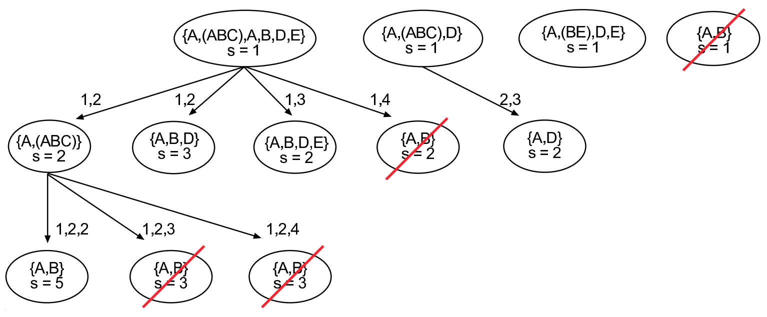


Figure 6.1: RERII with sequential transactions in Table 6.1

The intersection between node 1 ({A,(ABC),A,B,D}) and node 2 ({A,(ABC),D})

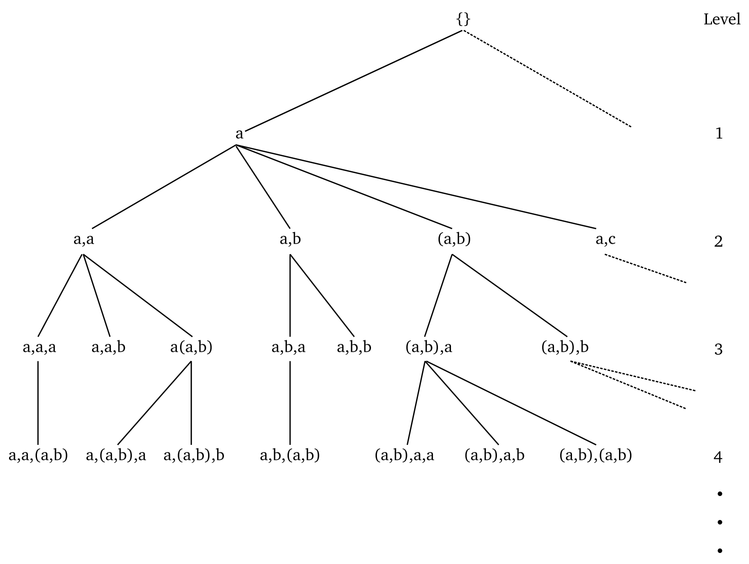


Figure 6.2: Snippet of sequential-item-enumeration tree generated by SPAM (Ayres et al., 2002).

generates the sequences {A,(ABC)} and {A,B,D}. RERII would consider both of these to be two sibling child nodes of {A,(ABC),A,B,D}. However, in the next enumeration {A,(ABC)} and {A,B,D} will be intersected, generating the child node {A,B} with a support of 3, which is eventually increased to 5 after {A,(ABC)} is intersected with {ABDE} and {AB}. This arises from taking the intersections between the nodes 1 and 2 more than once. This is incorrect – the maximum support any sequence can have is bounded by the initial number of sequential transactions, which is four in this example.

SEQRE overcomes this issue by altering the process of constructing the enumeration tree. As SEQRE is best explained by example, we will continue to refer to Figures 6.3, 6.4 and 6.5 based on the transactions in Table 6.1 to illustrate our approach.

6.2.1.1 Tree Representation

In SEQRE we represent each node N of the sequential row enumeration tree with a two element group N = {sequenceList, childMap}, where the sequenceList is a set of *closed sequences* (Definition 25) that are generated by intersecting the parent node of N with one of its siblings. Each closed sequence has an individual support count assigned to it. The childMap is simply a list of links to all child nodes. For example, the node formed by the intersection between {A,(ABC),A,B,D} and {A,(A,B,D),D} can be represented with { {A,(A,B,C) - 2 : A,B,D - 2} {A,B - 3}}.

6.2.1.2 Child Node Construction

Constructing the initial child nodes of the transaction nodes is straightforward (Figure 6.3, steps 1 - 4). The construction of the next generations of nodes is more complicated and involves the following steps:

1. Construct Dummy Child Nodes
Given a node X with sequenceList x_1, x_2, ..., x_i and a sibling node Y with sequenceList y_1, y_2, ..., y_j, we will create individual dummy child nodes t_1, t_2, ... t_i, based on the following properties:

1. if $x_1 \cap y_1 = \emptyset$ no dummy nodes need to be constructed.
2. if $x_1 \subseteq y_1$, then
   - if $|X| = 1$, y_1 is pruned/integrated into the sequence x_1 and the support of x_1 is increased, as well as any of x_1's child nodes.
   - else y_1 becomes a dummy node of x_1, with increased support.
3. if $y_1 \subset x_1$, then $x_1 \cap y_1$ becomes a dummy node of x_1 with increased support if it has not been discovered previously.
4. if $x_1 \neq y_1$, then $x_1 \cap y_1$ becomes a dummy node of x_1 with increased support if it has not been discovered previously.

Note that the support of x_i can only increase by 1, even if it is contained within more than one sequence of Y.

In Figure 6.3(e), dummy nodes are drawn as boxes. The dummy node {A,B -3} is formed by intersecting {A,(A,B,C)} with {A,B,D,E} – Property 3.

In Figure 6.4(a), when the sequence {A,B,D} is intersected with {A,B,D,E}, it is found that {A,B,D} ⊂ {A,B,D,E}. When this is the case in RERII, the child node {A,B,D} will not be created, and the parent node {A,B,D} support will be increased. In SEQRE, only if the parent node has one sequence is its support increased without child node expansion. This is not the case here, and the dummy node {A,B,D - 3} is created – Property 2.

2. Construct Real Child Node
After all dummy child nodes of a node *X* and its sibling node *Y* are constructed they are merged to form a single *real child node*. Any sequence of a dummy node which is a subsequence of another is pruned to maintain the closure property. In Figure 6.4(b) the dummy nodes {A,B} and {A,B,D} are merged to form the real child node {A,B,D}. The dummy node {A,B} is pruned indicated by red stroke. This real child node corresponds to the subsequence that is common between transactions 1, 2 and 3, and thus has a support of 3.

Once all real child nodes of *X* are generated, the algorithm continues in a depth-first manner (Figures 6.5(a) and 6.5(b)), until no child nodes can be formed.

6.2.1.3 Maximal Sequential Pattern Pruning

Mining all frequent closed sequential patterns will extract many redundant sequential patterns. In mining microarray data, the number of patterns found with suitable support can be enormous. This will restrict a biologist's understanding as they attempt to navigate through and comprehend the sequential pattern space.

Ideally we would like to reduce the number of sequential patterns generated as much as possible, without any loss of information to aid in a biologist's interpretation of the results.

We observed that by searching for *Maximal Sequential Patterns* we do not lose any information whilst reducing the number of patterns found.

Definition 28 (Maximal Sequential Patterns) *Let $\mathcal{F}$ be the set of frequent sequential patterns in a sequential dataset D. The set M of* maximal sequential patterns *in D is the set of frequent sequential patterns whose super-patterns are infrequent, formally*

$$M = \{P_1 \in \mathcal{F} \mid \nexists\, P_2 \in \mathcal{F},\, P_1 \subset P_2\} \tag{6.4}$$

For example if the pattern $x = \{$A,(BCD),E,F$\}$ is not frequent, but the pattern $y = \{$A,(BC)E,F$\}$ is, than y is a maximal pattern.

Traditional Apriori style methods introduced in Chapter 4 do not benefit from mining maximal frequent patterns. A frequent pattern is considered maximal if none of its immediate super-patterns are frequent. However, at any point in the Apriori algorithm we only know the support of a sequential pattern's sub-patterns and thus we cannot determine if it is maximal until we generate and tests its super-patterns. Therefore, in the process of generating maximal sequential patterns, all frequent closed patterns will be generated. Thus there is no computational advantage for Apriori methods to mine maximal sequential patterns.

Maximal sequential patterns can be mined using a our algorithm SEQRE efficiently by successfully generating all maximal sequential patterns without considering any of their sub-patterns. SEQRE can exploit the *support downward closure* property, defined in Lemma 2. This property allows us to effectively prune all subpatterns of a frequent sequential pattern when discovered, enabling the algorithm to return from that depth first iteration. When a frequent sequential pattern is the first to be discovered down a depth first path, it is a maximal sequential pattern, of all the generations of patterns that may form below it. Based on this, SEQRE will never consider any subpatterns of maximal sequential patterns. For example if we set the support threshold to 3, no further enumeration is required below node $\{$A,B,D$\}$ in Figure 6.4(c) as all possible child nodes are guaranteed to be frequent.

Lemma 2 (Support Downward Closed) *For a given frequent sequential pattern of size ≥ 2 with support s, the support of its subpatterns are bounded below by s. Thus if we identify a frequent sequential pattern P, all of its subpatterns are redundant in that we can approximate their support from P.*

Proof 2 *Let $\mathcal{F}$ be the set of frequent sequential patterns. Let $P_1 \subseteq P$, $P_1 \in \mathcal{F}$ and $P_2 \subset P_1$, then*

$$\begin{aligned} P_2 \subset P_1 &\implies support(P_2) \geq support(P_1) \geq min\_support \\ &\implies P_2 \in \mathcal{F} \end{aligned}$$

This approach will make mining maximal sequential patterns more efficient than algorithms searching for frequent closed sequential patterns. Another advantage of this top-down approach is that we do not lose any information. We are guaranteed that all subpatterns of a maximal sequential pattern have at least the support of the maximal sequential pattern, and thus we effectively reduce the itemsets identified without significant information loss.

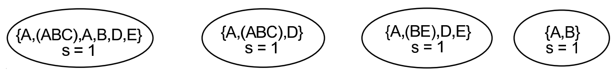
6.3 SEQRE Analysis

We applied our proposed SEQRE algorithm to the yeast cell cycle temporal microarray data collected by Spellman et al. (1998). This dataset contains 4 separate cell-cycle temporal experiments, each monitoring the expression level of 6177 genes over approximately 20 time frames, totally to 76 individual time frames. As with our previous approach, we treat each gene as two individual items, one for its up-regulation and the other for its down-regulation.

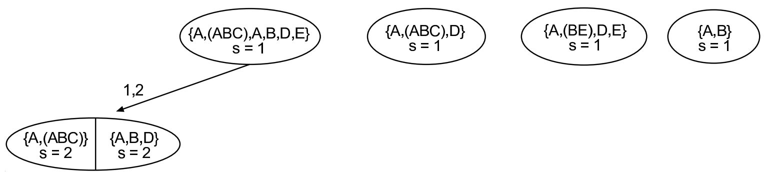
In this section we investigate the performance of SEQRE. We systematically analyse our algorithm's complexity issues and general effectiveness in generating patterns which can be supported by existing biological experiments. As no prior algorithms have been designed to analyse microarrays globally, it is difficult to compare our results to others.

6.3.1 Computational Effectiveness

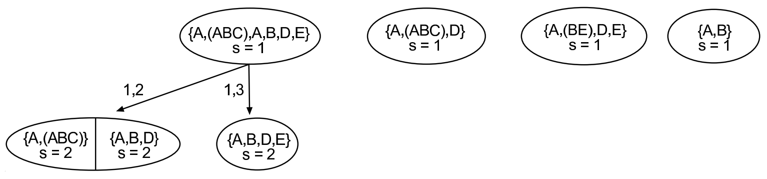
Table 6.2 shows the significant reduction in the number of patterns generated with and without the extra support pruning. As such, the time required to generated the



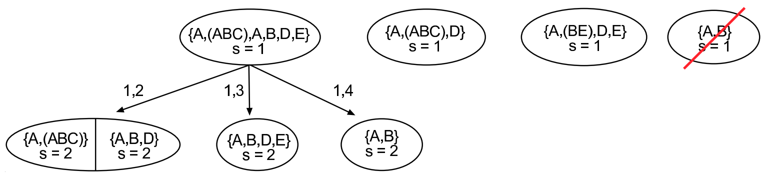
(a) Initialisation



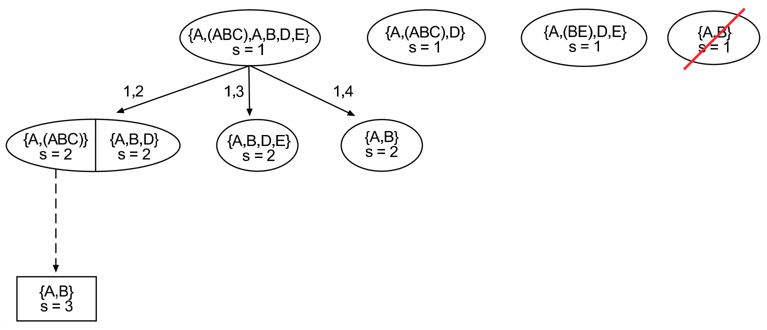
(b) Step 1



(c) Step 2

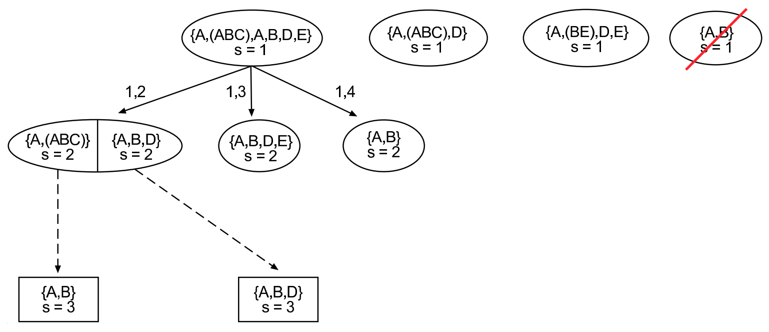


(d) Step 3

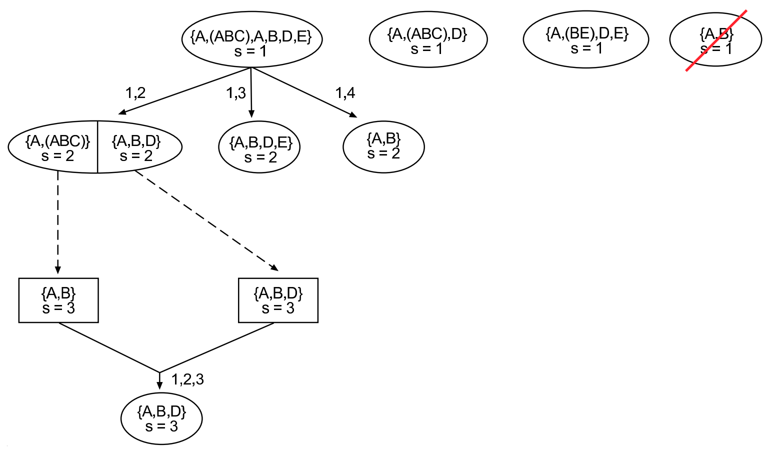


(e) Step 4

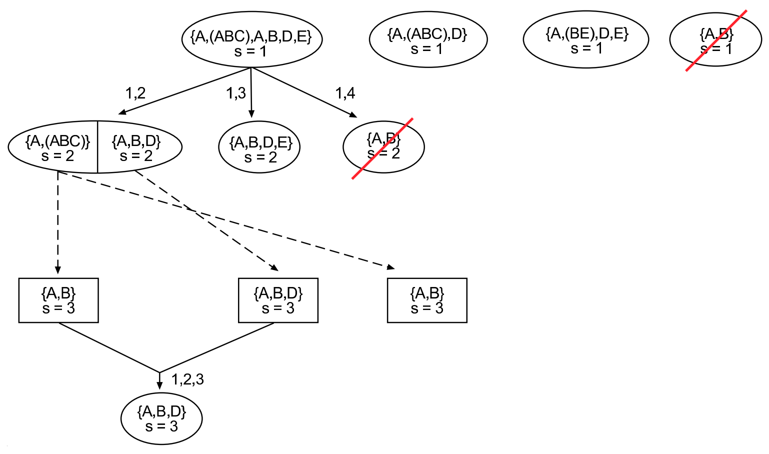
Figure 6.3: SEQRE algorithm example - Part 1 (Min Support = 2)



(a) Step 5

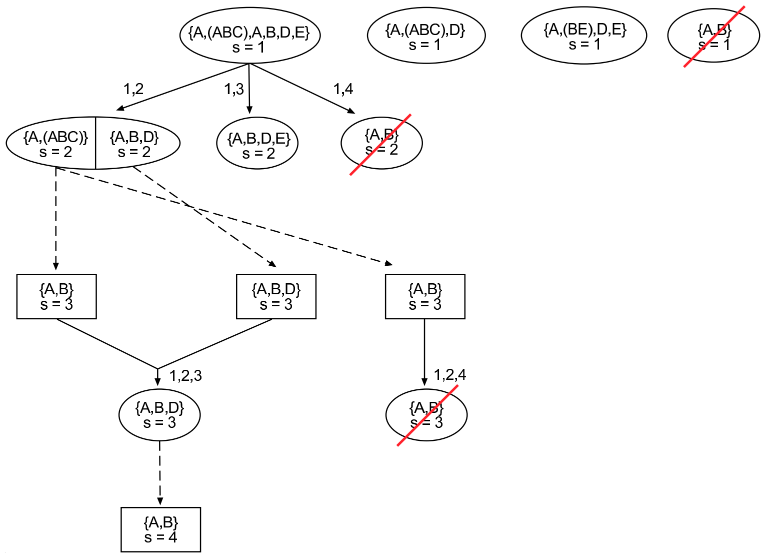


(b) Step 6

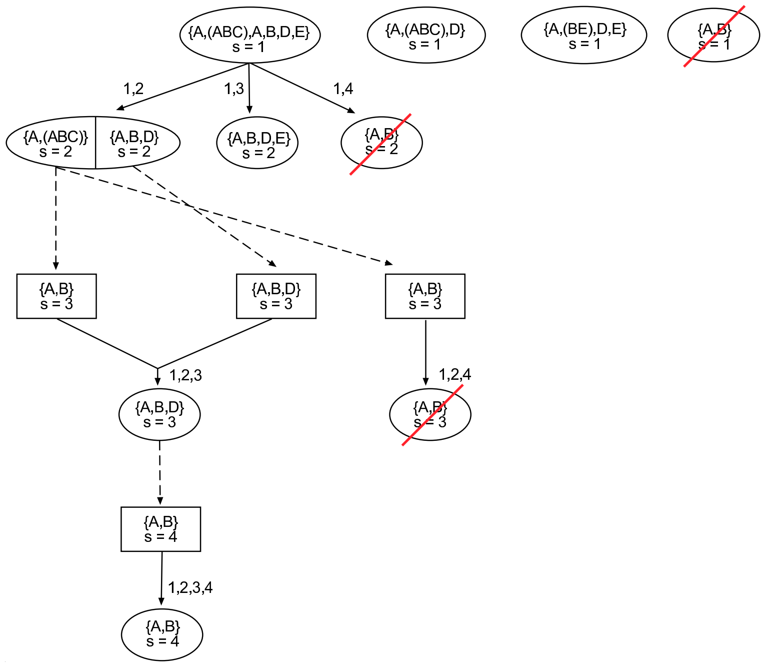


(c) Step 7

Figure 6.4: SEQRE algorithm example - Part 2 (Min Support = 2)



(a) Step 8



(b) Step 9

Figure 6.5: SEQRE algorithm example - Part 3 (Min Support = 2)

Support(%)	# Patterns	
	Closed	Maximal
10	546869	114550
20	440235	88217
30	45161	20432
40	3149	2015
50	155	133

Table 6.2: Effect of support closure pruning for pattern extraction

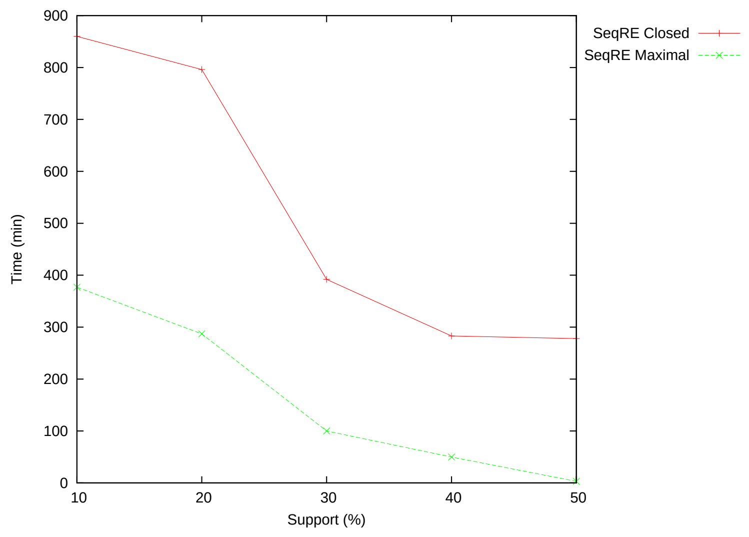


Figure 6.6: Scalability of SEQRE

maximal sequential patterns, is significantly less, highlighted in Figure 6.6.

To test the correctness of SEQRE we applied SEQRE to random datasets. Each dataset was created by randomly permuting the order of the time series for each gene independently. This maintains the composition of the time series, while inducing independence between the genes. From such datasets it is not expected that many patterns will be identified or exhibit true biological relationships. As expected our approach extracted significantly less patterns on average, with no patterns with support above 40% found (Table 6.3). From this we conclude that many of the patterns we find from the real dataset are not spurious.

Support(%)	# Patterns	
	SeqRE Maximal	Random[a] Maximal
10	114550	12728
20	88217	6107
30	20432	529
40	2015	0
50	133	0

[a]average of 100 random repeats

Table 6.3: Robustness of SeqRE for pattern extraction

6.3.2 Biological Analysis

In this section we confirm our the ability of our approach to extract sensible patterns between genes of known function. From our analysis we conclude that our approach is capable of extracting patterns indicative of biological phenomena in the data.

#	Pattern	Support(%)
1	{DBP3}, {$\overline{\text{CDC20}}$}, {$\overline{\text{ASH1}}$}, {NUM1, CLB1, CLB2, $\overline{\text{TEC1}}$, $\overline{\text{ASH1}}$, $\overline{\text{CDC46}}$ }, {SWI5, CLB1, CDC5, CLB2}	50
2	{NUM1},{CLB1 CDC5}, {CDC47, SWI5, CDC20, CLB2, $\overline{\text{POL12}}$, $\overline{\text{MCD1}}$, $\overline{\text{RAD54}}$, $\overline{\text{CDC45}}$, $\overline{\text{CLN2}}$, $\overline{\text{DPB2}}$ }, {CDC47, DBF2, FAR1}, {RME1, ASH1, CDC46, EGT2}	25
3	{$\overline{\text{SWI5}}$}, {CTS1}, {MNN1, CLN1}, {$\overline{\text{ASH1}}$}, {$\overline{\text{KAR4}}$, $\overline{\text{ASH1}}$, $\overline{\text{EGT2}}$,$\overline{\text{AGA1}}$}	27
4	{CDC42}, {STE20}, {$\overline{\text{DIG1}}$ }, {$\overline{\text{FUS1}}$, $\overline{\text{FUS3}}$, $\overline{\text{STE12}}$}	52

Table 6.4: Example patterns extracted by SeqRE

Original clustering and manual analysis of this dataset by Spellman et al. (1998) identified 800 genes whose expression varied over the course of the experiments. These genes were considered to be cell-cycle regulated.

By inspection of the pattern sets generated by SeqRE, we observed the existence of many *dominant genes*. Dominant genes are those that are strongly overrepresented within the patterns and include the genes MCD1, RFA2, CDC45, CLN2, POL30 and RAD53. A similar discovery was made by Friedman et al. (2000) who applied Bayesian networks to the same dataset analysing only the 800[1] genes previously confirmed to be cell-cycle regulated. They concluded that the dominate genes are the main genes directly involved in the initiation and the control of the cell-cycle (Friedman et al., 2000). This is not a surprising result considering that Friedman et al. (2000) only analysed cell-cycle genes. With respect to our approach, we consider that extracting such dominant gene relationships, from the thousands of genes we analyse, is a positive result. The other temporal relationships we identify that contain genes not considered cell-cycle regulated, may warrant further investigation. Identifying relationships not involving all cell-cycle genes is not surprising – the cell-cycle is not the only cellular process that may be occurring. For example pattern 4 in Table 6.4 correctly describes the *MAPK Signalling Pathway* induced during nutrient starvation.

Furthermore, the Bayesian approach of Friedman et al. (2000) cannot include the expression changes in two or more consecutive time points, whereas we have applied our approach to all time points in one process successfully.

6.4 Gene Network Construction

Standard association rule mining methods applied to low density data typically generates a large number of rules. Closed itemset generation and other measures of *interestingness* are often incorporated into the mining process to reduce this rule set without information loss. Furthermore, rule compression and rule clustering methods (Gupta et al., 1999; Lent et al., 1997) can be applied to reduce the rule set. However, the size of the rule set is still often impossible for users to comprehend. This is certainly the case for microarray data.

The process of generating gene networks from the gene relationships extracted from Boolean and Bayesian approaches is simple due to the limitations on the

[1]Here many individual Bayesian networks were constructed as only a small select group of genes could be studied in each

number of genes that are analysed. The entire Boolean and Bayesian networks can be drawn clearly and quickly. However, with the limited input its output is too limited. Therefore, if information regarding a gene that was not within the selected group studied is requested at a later time, an entirely new network must be constructed. This is one of the major downfalls of not analysing microarrays globally.

Unlike existing methods, MAXCONF and SEQRE extract relationships on a global scale. Unfortunately this makes it impossible to view and understand the entire network of relationships. With this in mind we propose a *Local Gene Network* inspired by the *Association Rule Network*, which provides an effective method for navigating and visualising the large rule space.

6.4.1 Association Rule Network

The recently proposed *Association Rules Network* (ARN) (Chawla et al., 2004) allows one to understand the itemsets that are frequently associated with a selected item of interest visually. They provide a method to see many of the relationships that span from a goal item of interest. An ARN is a directed acyclic hyper-graph with no backward edges or redundant paths which captures the relationships between items leading up to the selected item. The nodes correspond to frequent items and the hyper-edges correspond to rules whose consequent has a cardinality of one. An ARN is generated by recursively joining antecedent items that appear in rules together with the current goal item as a single consequent with a hyper-edge. Each hyper-edge is assigned a weight - the confidence of the respective rule. Following construction, all cycles and reverse edges are pruned.

Figure 6.7(a) shows an ARN generated from the rules in Table 6.5 with goal item *E*. Both items *C* and *D* are antecedents when E is a single consequent, and thus are connected to *E*. Due to limitations in the ARN model, the node corresponding to item *C* is only linked back to the item *A*, even though *C* also appears as a consequent in Rule 2. This is discussed further in Section 6.4.2.

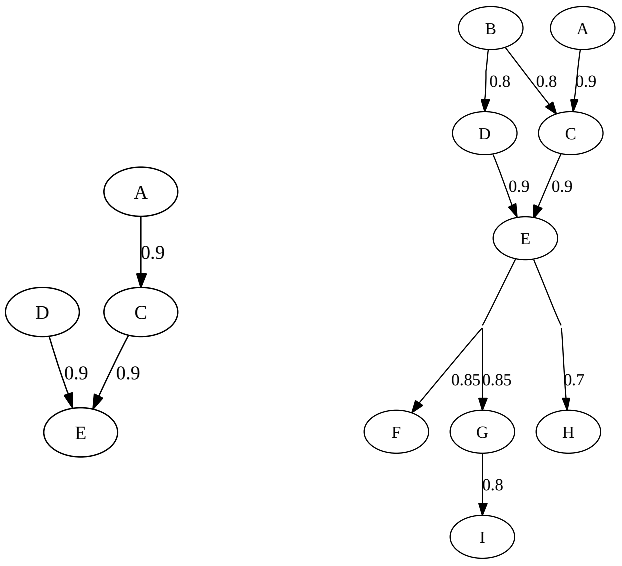
#	Rule	Confidence
1	A ⇒ C	0.9
2	B ⇒ C,D	0.8
3	C,D ⇒ E	0.9
4	E ⇒ F,G	0.85
5	E ⇒ H	0.7
6	G ⇒ I	0.8

Table 6.5: Example association rules

6.4.2 Local Gene Network

ARNs appear ideal for modelling gene networks from association rules providing biologists with a method for examining the gene relationships concerning a particular gene of interest. ARNs exhibit many structural similarities to gene networks, such as logical *AND* (hyper-edges spanning more than two nodes) and *OR* (consequents with multiple hyper-edges connecting their antecedents) relationships between genes like Boolean networks and provide a statistic of how well the network models the observed expression data (edge confidence weight) similar to that of the Bayesian network. However, from a biological perspective ARN generation suffers from four main structural restrictions, each of which reduces the knowledge one may extract about a goal item.

1. The goal item must appear as a singleton consequent in the rule set – this potentially ignores many relationships, and is the case for *C* in rule 2
2. The network is only generated backwards from the goal item. No forward relationships are depicted
3. No cycles are allowed – cycles can reveal important biological mechanisms such as *Feed Forward Loops* and *Feed Backward Loops* (Milo et al., 2002).
4. No negative relationships – in the case of microarray data we wish to know genes which activate and inactivate the goal gene and which genes the goal gene may activate or inactivate.



(a) Association Rule Network (b) Local Gene Network

Figure 6.7: Example ARN and LGN with goal node E and corresponding hyper-edge confidence values, reflecting item relationships from rules in Table 6.5.

We propose a *Local Gene Network* (LGN) to model the local relationships of genes with a gene of interest, which overcomes the restrictions of the ARN, providing all possible relationships. Despite the following definitions and properties of LGNs being introduced in the context of association rules, the LGN is not restricted to such relationships. Each property can be easily extended for sequential relationships, as each *k*-pattern can be treated as a set of *k* rules.

Definition 29 (Local Gene Network) *Given a set of association rules R describing relationships between genes and a frequent goal gene z, which appears in a antecedent or consequent of a rule $r \in R$, a* local gene network*, LGN(R, z) is a weighted graph such that*

1. *There is a hyper-edge which corresponds to a rule r_0 whose consequent contains the gene z and/or a rule r_1 whose antecedent contains the gene z.*

2. *Each hyper-edge in LGN(R,z) corresponds to an association rule R local to*

z.

3. *Each hyper-edge in LGN(R,z) can represent one of four relationships, arising from the four types of rules.*
   (a) *Positive activating $A \Rightarrow B$*
   (b) *Negative activating $\overline{A} \Rightarrow B$*
   (c) *Positive repression $A \Rightarrow \overline{B}$*
   (d) *Negative repression $\overline{A} \Rightarrow \overline{B}$*
4. *The weight of each hyper-edge in LGN(R,z) is the confidence of it's corresponding rule.*
5. *Any gene $i \notin$ LGN is not local from gene z.*

Definition 30 (Local Association Rule) *An association rule $r \in R$, is* local *to the goal item if their exists a forward or backward path of length L from the goal to the antecedent or consequent of r. The length L is the preselected maximum distance item nodes can be from the goal.*

The Local Gene Network in Figure 6.7(b) was generated from the rules in Table 6.5 and depicts the usefulness of LGNs compared to ARNs. An obvious improvement of the LGN is the incorporation of the item *B* which is not contained in the ARN as the item C is not a single consequent in Rule 2. Furthermore LGNs successfully display all *forward* relationships from *E* (Rules 4 and 5 in Table 6.5).

We constructed Local Gene Networks from the association rules we discovered using MaxConf and SeqRE, four of which are shown in Figures 6.8, 6.9, 6.10 and 6.11. These gene networks can be interpreted as follows:

1. Black Full arrows represent Positive activating relationships
2. Black Dashed arrows represent Negative activating relationships
3. Red Full Barred arrows represent Positive repression relationships
4. Red Dashed Barred arrows represent Negative repression relationships

6.5 Summary

This chapter presented the first top-down algorithm for sequential pattern mining. We show that our approach efficiently mines sequential patterns from temporal microarray data, incorporating all genes and time measurements. This is a significant improvement to existing algorithms. An evaluation of the sequential patterns we extract is provided indicating the strengths of our approach in identifying biological relationships. This chapter concludes by introducing our LGN model which effectively provides a tool to visualise and navigate the complex relationships we identify, some of which are shown in Figures 6.8 - 6.11.

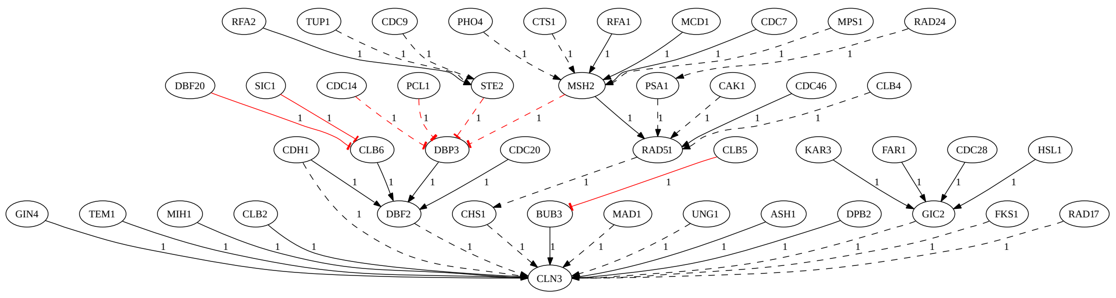


Figure 6.8: LGN with goal gene CLN3 where distance = 3

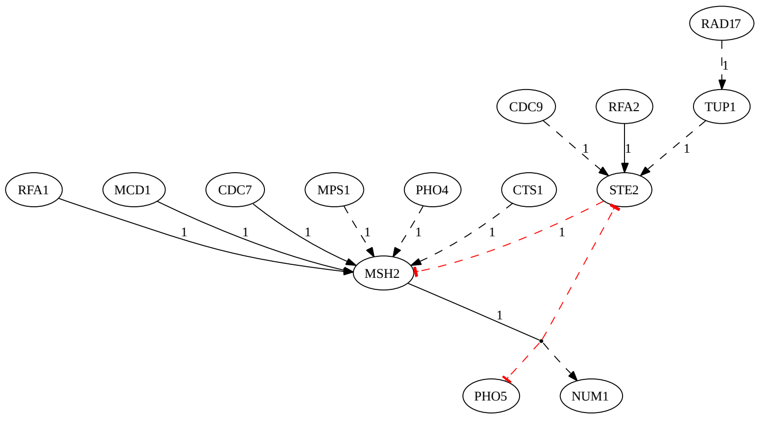


Figure 6.9: LGN with goal gene STE2 where distance = 3

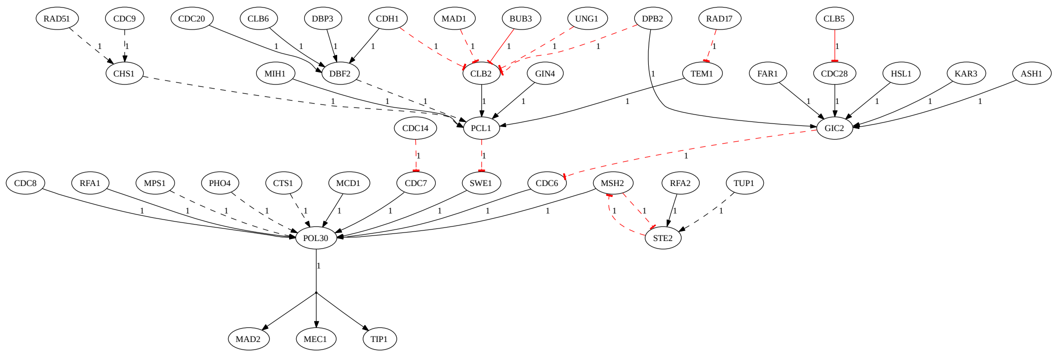


Figure 6.10: LGN with goal gene POL30 where distance = 3

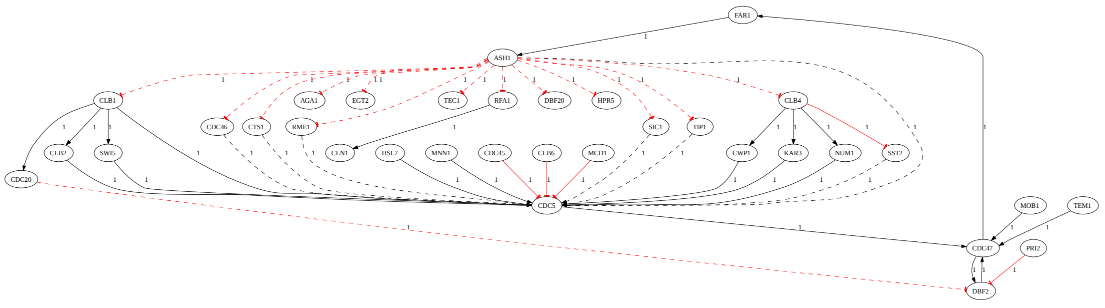


Figure 6.11: Sequential LGN with goal gene CDC5 where distance = 3

Algorithm 5 SEQRE - maximal sequential pattern mining - Part 1

```
Input: Min_support, Dataset $\mathcal{D}$ of sequential transactions
for all transactions $t \in \mathcal{D}$ do
    Remove all infrequent items from $t$
    N := $\emptyset$
    n := new Node(patterns = t.pattern, support = 1)
    N.append(n)
P := $\emptyset$ // Maximal Frequent Patterns
Discovered : = $\emptyset$ // Frequent Patterns idenfied
SEQRE_depthfirst(N, P)
Procedure: SEQRE_depthfirst(N, P)
for all node $n_i$ in $N$ do
    Children := $\emptyset$ // Real child nodes.
    for all node $n_j$ in $N$ where $n_j > n_i$ do
        dummy_Children : = $\emptyset$
        for all pattern $p_i$ in $n_i$ do
            dummy_$p_i$ : = $\emptyset$ // Dummy child nodes of $p_i$.
            $p_i$.incrementSup:= FALSE
            if $p_i$ can not be frequent then
                continue
            for all pattern $p_j$ in $n_j$ do
                intersections := getIntersection($p_i$, $p_j$)
                if intersections $\neq \emptyset$ then
                    call SEQRE_pruning_growth($p_i$,$p_j$)
            if $p_i$.support $\geq$ min_support then
                P.append($p_i$)
                // Support downward closed - No need to expand children
                dummy_$p_i$ : = $\emptyset$
            else
                dummy_Children.append(dummy_$p_i$)
        // Construct Real Children
        child := mergeDummies(dummy_Children)
        Children.append(child)
    if Children $\neq \emptyset$ then
        call SEQRE_depthfirst(Children, P)
```

Algorithm 6 SEQRE - maximal sequential pattern mining - Part 2

```
Procedure: SEQRE_pruning_growth($p_i$,$p_j$)
if $p_i \subseteq p_j$ then
    delete $p_j$
    if $|n_i| \geq 1$ then
        dummy := new Node($p_i$.sequence, $p_i$.support + 1)
        dummy_$p_i$.append(dummy)
    else if $p_i$.incrementSup == FALSE then
        $p_i$.support++
        $p_i$.incrementSup = TRUE
    if $p_i \supset p_j$ then
        if $p_i \cup p_j$ not discovered before then
            dummy := new Node($p_i$.sequence, $p_i$.support + 1)
            dummy_$p_i$.append(dummy)
    if $p_i \neq p_j$ then
        if $p_i \cup p_j$ not discovered before then
            dummy := new Node($p_i$.sequence, $p_i$.support + 1)
            dummy_$p_i$.append(dummy)
```

Algorithm 7 Local Gene Network

```
Input: Rules $R$, Target Gene, maxLocal, Network $N$
visitedCons[i] = 1 if $i$ has been visited as a consequent
visitedAnte[i] = 1 if $i$ has been visited as a antecedent
call build_LGN(target, maxLocal)
call build_LGN(-target, maxLocal)
Procedure: build_LGN(target,local)
if local $< 0$ then
   return
/* Get all rules which have target as an antecedent */
RuleSubset := Rules.getRulesAnte($R$, target)
for all rule $r \in RuleSubset$ do
   for all antecedent $\in r$ do
      if visitedAnte[antecedent] = 0 then
         /* Directed edge from antecedent to target with weight confidence */
         if target = negative then
            $N$.add_edge(antecedent, target, r.confidence, inactivate)
         else
            $N$.add_edge(antecedent, target, r.confidence, activate)
         call build_LGN(antecedent, local-1)
/* Get all rules which have target as a consequent */
RuleSubset := Rules.getRulesCons($R$, target)
for all rule $r \in RuleSubset$ do
   for all consequent $\in$ r do
      if visitedCons[consequent] == 0 then
         /* Directed edge from target to consequent with weight confidence*/
         if consequent = negative then
            $N$.add_edge(target, consequent, r.confidence, inactivate)
         else
            $N$.add_edge(target, consequent, r.confidence, activate)
         call build_LGN(consequent, local-1)
```

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Microarrays have the potential to revolutionise the way in which biological research is carried out. One main objective of molecular biologists is to develop a deeper understanding of how different cells control and regulate the expression of their genes. These mechanisms can be depicted in gene networks. However, each microarray experiment generates an enourmous amount of information for which biologists have no effective means to explore and therefore many gene relationships remain hidden.

Existing methods for extracting gene networks from microarrays suffer from two main short-comings. They significantly restrict the search space to only a subset of genes and consecutive time measurements. This is because microarray data is dense, rendering many data mining techniques infeasible. Recently top-down association rule mining methods have emerged to facilitate the mining of microarray data. These approaches can consider all the genes in a microarray experiment. However, they are still limited in the information they can extract. These algorithms rely on the support measure to restrict the search space, leaving *many potentially interesting* relationships which have low support and high confidence undiscovered.

In this thesis we presented the first comprehensive confidence based top-down algorithm MAXCONF which is capable of identifying interesting gene relationships on a global genome scale. We show that our approach efficiently identifies high confidence rules without support pruning. Further we demonstrate the biological significance of the relationships we identify. The main result of our evaluation against existing methods based on support pruning indicates that we achieve a significant recall improvement with an increase from 0.15% to 94%.

We also presented the first top-down algorithm specifically designed to mine sequential patterns from microarrays without any search space restrictions. SEQRE is the only algorithm to date that is capable of handling the complexity of temporal microarray data. SEQRE efficiently mines sequential patterns from temporal microarrays considering all consecutive time frames and all genes. This is possible due to the extra pruning property of support we identified, which can only be exploited in top-down sequential pattern mining algorithms. Our evaluation of SEQRE demonstrates that it identifies many biological relationships.

Both of the algorithms MAXCONF and SEQRE presented in this thesis are shown to identify thousands of gene relationships, which would be impossible for a biologist to interpret. We propose the *Local Gene Network* to provide a network visualisation to aid navigation through this large relationship space.

In this thesis, we have provided complete, principled and efficient solutions for the mining, integration and analysis of microarrays. Each of our approaches will benefit the biological community tremendously by finally providing them with effective tools to analyse and extract gene networks from microarrays.

7.2 Future Work

An important open area for future work is to combine the relationships identified from both Knockout and Temporal microarray experiments. Both types of microarrays provide a wealth of very different, detailed information about the genes studied. Therefore, a method to combine all gene relationships will produce a significantly more detailed gene network, providing biologists with as much information as possible. An algorithm which can take both forms of data simultaneously would be very interesting and challenging to develop.

Another direction our work may take is to extend our algorithms to allow incremental network construction. This approach is inspired by the way in which biologists carry out their microarray experiments. Biologists do not typically carry out all microarray experiments at once. Often a biologist decides to carry out more experiments to test new hypotheses driven from previous experiments. With this method it is expected that over time a large library of microarrays for a single organism will become available. This situation is ideal - the more microarrays that are available for network construction, the more statistically sound the gene relationships identified will be. However, a biologist should not need to wait until the completion of all experiments to extract gene relationships. Therefore it would be of interest to modify our algorithms to allow incremental network reconstruction, so to refine the gene relationships when new data becomes available.

Finally, our work will not only benefit those within the biological community. Our research not only introduces new techniques for microarray analysis. The data mining ideas behind both our MAXCONF and SEQRE algorithms are directly applicable to other dense datasets. It would be interesting to see how our approaches perform on such datasets.

Bibliography

Dana-Farber Cancer Institute - Microarray Core Facility, Harvard Medical School, Boston, Massachusetts, USA. http://chip.dfci.harvard.edu/stats/, 2005.

The Legume-Microbe Interactions Laboratory, The University of Missouri, USA. http://www.psu.missouri.edu/staceylab/signal.htm, 2005.

R Agrawal, T Imielinksi, and A N Swami. Mining association rules between sets of items in large databases. In *The 1993 ACM SIGMOD International Conference on Management of Data*, volume 22, pages 207–216, 1993.

R Agrawal and R Srikant. Mining sequential patterns. In *Int'l. Conf. Data Engineering (ICDE 95)*, pages 3–14, 1995.

T Akutsu, S Kuhara, O Maruyama, and S Miyano. Identification of genetic networks by strategic gene disruptions and gene overexpressions under a boolean model. *Theoretical Computer Science*, 298:235–251, 2003.

T Akutsu, S Miyano, and S Kuhara. Indentification of genetic networks from a small number of gene expression patterns under the boolean network model. *Pacific Symposium on Biocomputing*, 4:17–28, 1999.

T Akutsu, S Miyano, and S Kuhara. Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, 16(8):727–734, 2000.

C Antunes and A.L Oliveria. Sequential pattern mining algorithms: trade-offs between speed and memory. In *PKDD Workshop on Mining Graphs, Trees and Sequences*, September 2004.

J Ayres, J Gehrke, T Yiu, and J Flannick. Sequential pattern mining using a bitmap representation. In *Int'l Conf Knowledge Discovery and Data Mining*, pages 429–435, 2002.

P Baldi. *DNA microarrays and gene expression: from experiments to data analysis and modeling*. Cambridge University Press, Cambridge, 2002.

T Beissbarth and T.P Speed. Gostat: Find statistically overrepresented gene ontologies within gene groups. *Bioinformatics*, 20(9):1464–1465, 2004.

C Alfarano et.al. The Biomolecular Interaction Network Database and related tools 2005 update. *Nucleic Acids Res*, 33:D418–24, 2005.

S Chawla, J Davis, and G Pandey. On local pruning of association rules using directed hypergraphs. *IEEE ICDE*, 2004.

G Cong, K-L Tan, A Tung, and F Pan. Mining frequent closed patterns in microarray data. In *IEEE ICDM*, volume Fourth, pages 363–366, 2004.

C Creighton and S Hanash. Mining gene expression databases for association rules. *Bioinformatics*, 19(1):79–86, 2003.

M. B Eisen, P.T Spellman, P.O Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Genetics*, 95:14863–14868, December 1998.

N Friedman, M Linial, I Nachman, and D Pe'er. Using bayesian networks to analyze expression data. *Journal of Computational Biology*, 7:601–620, 2000.

N Friedman, K Murphy, and S Russell. Learning the structure of dynamic probabilistic networks. *Pacific Symposium on Biocomputing*, 8:17–28, 1999.

G Gupta, A Strehl, and J Ghosh. Distance based clustering of association rules. In ASME Press, editor, *Intelligent Engineering Systgems Through Artificial Neural Networks (Proceedings of ANNIE 1999)*, volume 9, pages 759–764, 1999.

Y Huang, H Xiong, S Shekhar, and J Pei. Mining confident co-location rules without a support threshold. In *Proceedings of the 18th ACM Symposium on Applied Computing ACM SAC*, 2003.

T.R Hughes et al. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–126, 2000.

M Kanehisa and S Goto. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res.*, 28(1):27–30, 2000.

B Lent, A.N Swami, and J Widom. Clustering association rules. In *ICDE*, pages 220–231, 1997.

B Lewin. *Genes VII*. Oxford University Press Inc., New York, 7th edition, 2000.

S Liang, S Fuhrman, and R Somogyi. Reveal, a general reverse engineering algortihm for inference of genetic network architectures. *Pacific Symposium on Biocomputing*, 3:18–29, 1998.

R Milo, S Shen-Orr, S Itzkovitz, N Kashtan, D Chklovskii, and U Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298:824–827, 2002.

F Pan, G Cong, K.K.H Tung, J Yang, and M.J Zaki. Carpenter: Finding closed patterns in long biological datasets. In *ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD)*, 2003.

N Pasquier, Y Bastide, R Taouil, and L Lakhal. Discovering frequent closed itemsets for association rules. In *Proceedings of the First IEEE Confernce on Data Mining*, pages 441–448, 1999.

J Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufman Publishers, INC, 2nd edition, 1988.

J Pei, J Han, and R Mao. Closet: An efficient algorithm for mining frequent closed itemsets. In *Proceedings of the 2000 ACM-SIGMOD International Workshop on Data Mining and Knowledge Discovery (DMKD '00)*, pages 21–30, 2000.

J Quackenbush. Microarrays - guilt by association. *Science*, 302:204–405, 2003.

F Rioult, J-F Boulicaut, B Cremileux, and J Besson. Using transposition for pattern discovery from microarray data. In *ACM-SIGMOD Int'l Workshop Data Mining and Knowledge Discovery (DMKD)*, 2003.

M.A Shipp et al. Diffuse large b-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. *Nature Medicine*, 8(1): 68–74, 2002.

I Shmulevich, E R. Dougherty, S Kim, and W Zhang. Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics*, 18(2):261–274, 2002.

Adrian Silvescu and Vasant Honavar. Temporal boolean network models of genetic networks and their inference from gene expression time series. *Complex Systems*, 13:54–70., 2001.

D.K Slonim, P Tamayo, J.P Mesirov, T.R Golub, and E.S Lander. Class prediction and discovery using gene expression data. In *4th Annual International Conference on Computational Molecular Biology (RECOMB)*, pages 263–272, 2000.

P.T Spellman, G Sherlock, M.Q Zhang, V.R Iyer, K.Anders, M.B Eisen, P.O Brown, D Botstein, and B Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9:3273–3297, 1998.

The Gene Ontology Consortium. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res*, 32:D258–D261, 2004.

Petri Toronen, Mikko Kolehmainen, Garry Wong, and Eero Castren. Analysis of gene expression data using self-organising maps. *Federation of European Biochemical Societies*, 451:142–146, 1999.

M.J Zaki and C Hsiao. Charm: An efficient algorithm for closed association rule mining. In *SIAM International Conference on Data Mining (SDM)*, 2002.