Knowledge Discovery in *Gene Expression* Databases

Siv. ing. Thesis

Knowledge Systems Group
Department of Computer and Information Science
Faculty of Physics, Informatics and Mathematics
Norwegian University of Science and Technology

Torgeir Rhodén Hvidsten

30.09.1999 – 02.03.2000

Abstract

This thesis is a contribution to the field of computational biology. It investigates what we will call knowledge discovery in *gene expression* databases. The need for computational methods and tools to analyse gene expression data is a result of the revolutionary new technology of microarrays. This technology is capable of measuring the gene expression level of thousands of genes in one single experiment and thus opens the possibility of doing large-scale gene research. The objective for this research is to reveal the biological functions of genes. In large scale, the function of ”unknown” genes can be inferred through their similarity to ”known” genes. Consequently, the problem is reduced to search the gene expression data sets for groups of related genes and to correlate these groups with existing biological knowledge. Moreover, the analysis can be further improved by inducing models that recognise these groups and that both give us knowledge about the relationship between genes and groups, and the capability of predicting the belonging of genes.

The task of searching data sets for natural underlying groups of related objects is known as unsupervised learning or clustering. Two conceptually different approaches to this problem exist. The first approach is what we call syntactical clustering. These methods search blindly for groups of related objects. In the thesis we propose a methodology for doing syntactical clustering of time series based on indiscernibility and preprocessing tools which include the Haar Wavelet Transformation and discretisation. The other approach is what we call semantical or knowledge-based clustering. These methods require a specification of what to search for in advance. One such approach is template-based clustering in that knowledge about the features of objects belonging to a specific cluster is encoded in templates. In the thesis we also propose a methodology for doing template-based clustering in time series.

The task of inducing models from data sets is called supervised inductive learning. In the biological domain these models need to meet the requirements of both predictive and descriptive nature. Pawlak’s rough set framework implemented in the ROSETTA system satisfies these requirements by inducing models consisting of a set of propositional rules.

A detailed investigation and comparison of the different clustering algorithms are provided and experiences using these methods and modelling on real world gene expression data sets are described. The work was done in close co-operation with biological experts and shows how powerful, and necessary, computational tools are in the analysis of gene expression data.

Acknowledgements

This thesis is the diploma-work of Torgeir Rhodén Hvidsten. The work was done at the Knowledge Systems Group, Department of Computer and Information Science, Norwegian University of Science and Technology. It is the result of co-operation and help from many people both at the Knowledge Systems Group and the Department of Physiology and Biomedical Engineering. I will first of all like to thank my supervisor Jan Komorowski for introducing me to the field and for creating a research environment that gave me both interesting discussions and much needed guidance, and to Astrid Lægreid for giving me the biological knowledge I needed and for spending hours interpreting my results. Thanks also to Arne Sandvik and Kristin Nørsett for sharing their biological knowledge. Then I would like to thank Tor-Kristian Jenssen and Dyre Tjeldvoll for our many discussions and for all their critical remarks. I would also like to thank the former for always solving the problem when my Perl programs refused to talk to me. Finally I would like to thank Aleksander Øhrn for his help on indiscernibility-based analysis and the ROSETTA system.

Thanks to Christin Grønnslett for not giving up on me when a spent all my time writing this thesis.

Contents

Chapter 1

Introduction

This work is a contribution to the field of computational biology. In particular it is concerned with what we will call knowledge discovery in gene expression databases. This chapter will give an introduction to the field both from a computer science and from a biological point of view.

Electronic equipment and low cost storage media have the last few decades made experts able to store large amounts of information. A large part of this information is primitive raw data. Since our ability to store data seems to exceed our ability to analyse it, a steadily increasing amount of this collected primitive raw data remains unanalysed. However, the reason why we collect and store this data is because we assume that it contain interesting information (knowledge). As a consequence, there is an urgent need for a new generation of computational theory and tools for extract knowledge from data. This new field within computers science is called *knowledge discovery in databases (KDD)*. The basic problem addressed by the KDD process is the one of mapping primitive raw data into a form that is more compact (a short report, a descriptive model, an approximation of the data, etc) or more useful (a predictive model for estimating the value of unseen cases).

DNA microarray technology is a classical example of how new methods and new technology require computational methods for storage and analysis of data. The method is based upon advanced robotic techniques printing high-density micro chips with DNA probes and analysis by fluorescence emission. It makes the scientists able to measure changes in the expression of thousands of genes in a single experiment and results in huge sets of gene expression data that needs to be analysed and correlated with existing knowledge. Methods from the KDD field seem like a natural tool in order to analyse gene expression data.

The objective of gene expression analysis is to find out which genes are responsible for which biological functions. This research is frequently refered to as *functional genomics*. A commonly used method for finding new mappings from genes to functions is to assume that genes with similar expressions are responsible for similar biological functions. Since some of the gene's functions are already known, the problem of finding gene-function mappings is reduced to finding genes with highly related expression patterns. If a subset of related genes have known functions, one might find it reasonable to suspect that the remaining unknown genes map to the same functions. The task of finding groups of similar objects in large data sets is known as cluster analysis or unsupervised learning in computer science. A variety of different statistical and mathematical methods already exist for this purpose. Having found the natural groups of genes in a data set, the analysis can be improved further by inducing a model that recognises these groups and thus can be used both to explain the relationships between genes and groups, and to predict the belonging of new genes. In some cases we have enough information about the known genes to classify them into groups without the help of clustering analysis. In these cases a model induced from the known genes can directly be used to predict the function of unknown genes. One approach in which modelling is used to predict the function of unknown genes is discussed in [Brown et al., 1999].

The aim of this thesis is to develop a method for computer-based gene expression clustering. The future target is to integrate this method with tools for modelling and evaluations, and with an information system that represents existing knowledge (see Figure 1.1). This will together result in a system that will become a powerful tool for any expert working with gene analysis. However, it is important to notice that any computer-generated result needs to be thoroughly interpreted by an expert in order to have a bio-medical value. The system is thus not an automation of the task of finding gene-function mappings, but rather an automation of the task of finding similar genes from a set of gene expression data. Our hypothesis is that the system described herein will make it easier for an expert to find gene-function mappings than it would be if he or she had to analyse the *raw* gene expression data. The main advantages of such a system can be captured in two clauses:

1. The expert can work with groups of similar genes [1] rather than with single genes. This will make the data more surveyable and will give valuable information about where to use time and money on more thorough experiments.

2. The expert has access to existing knowledge concerning the genes of current interest. This will become a major time-saving factor compared to searching for this information elsewhere and can thus be of great help in the search for a bio-medical interpretation of the clusters.

The system depicted in Figure 1.1 can be understood in an even wider context as shown in Figure 1.2. This figure shows how the bio-medical domain and computer science are brought together in order to accomplish the tasks of functional genomics. Data and knowledge are collected in a large data warehouse including expression data from microarray experiments, existing knowledge (re-) discovered from the WEB and new knowledge discovered through the KDD process.

Related work is done on similar systems as the one in Figure 1.1 and 1.2. [Moxon, 1998] discusses some of the same ideas, but no implementation has been done. Also work is being done at our group on collection, storage and correlation of existing knowledge. [Jenssen et al., 1999] and [Jenssen et al., 2000] both discuss the challenges of collecting knowledge from the WEB. In particular it investigates a method in which gene-gene-relations are mined from textual documents. [Tjeldvoll, 1999] discusses the challenge of collecting and representing biological knowledge, and to use this knowledge to validate analysed (clustered) gene expression data. Hence the work described in this thesis and the work described in [Jenssen et al., 1999] and [Tjeldvoll, 1999] should be seen in the context illustrated in Figure 1.1 and 1.2.

[1] This assumes that the system's definition of similarity between two genes is consistent with the expert's intentions. This, of course, demands a close co-operation with biological experts.

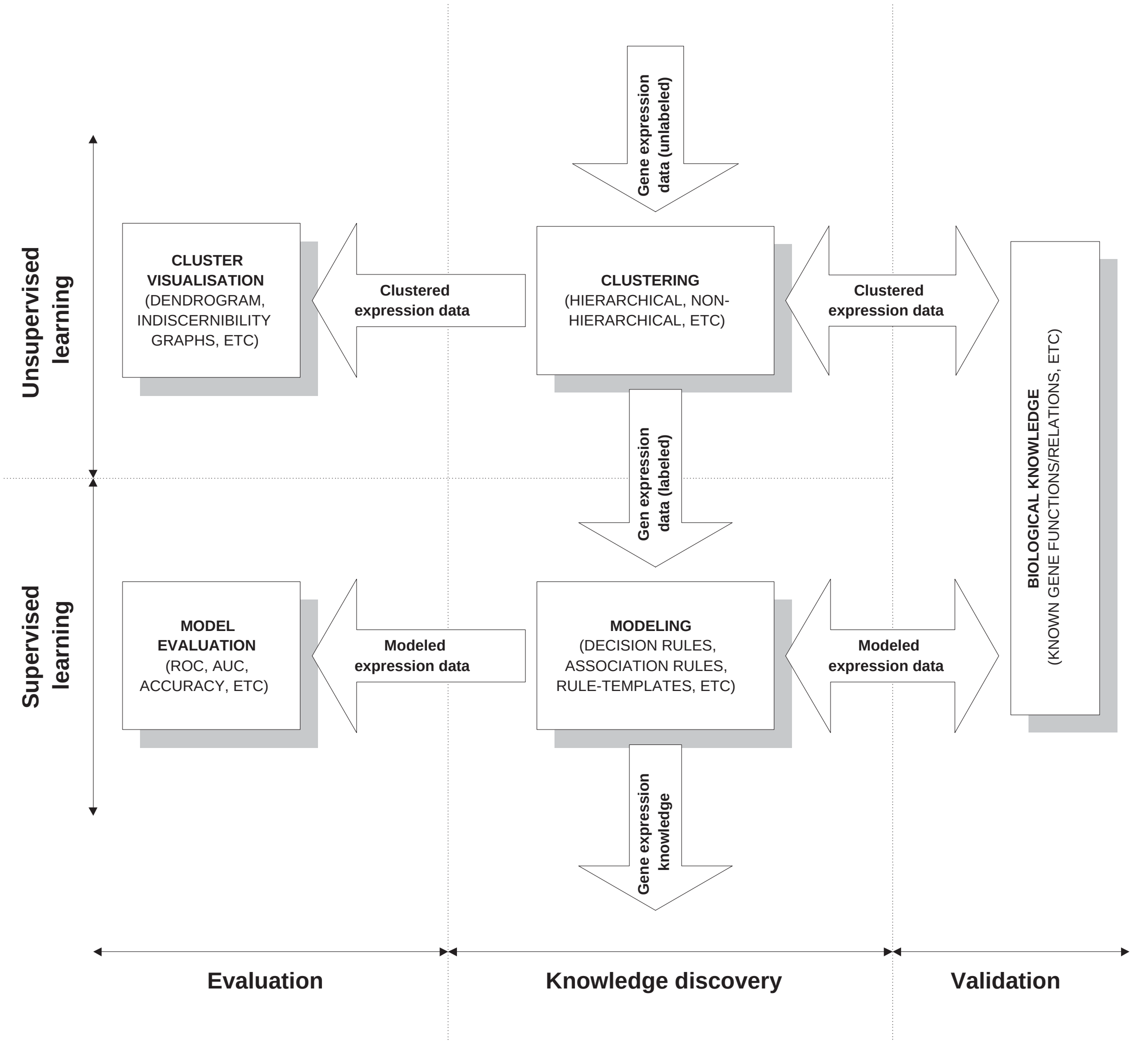

Gene expression data (unlabeled)
Unsupervised learning
CLUSTER VISUALISATION (DENDROGRAM, INDISCERNIBILITY GRAPHS, ETC)
Clustered expression data
CLUSTERING (HIERARCHICAL, NON-HIERARCHICAL, ETC)
Clustered expression data
BIOLOGICAL KNOWLEDGE (KNOWN GENE FUNCTIONS/RELATIONS, ETC)
Gen expression data (labeled)
Supervised learning
MODEL EVALUATION (ROC, AUC, ACCURACY, ETC)
Modeled expression data
MODELING (DECISION RULES, ASSOCIATION RULES, RULE-TEMPLATES, ETC)
Modeled expression data
Gene expression knowledge
Evaluation
Knowledge discovery
Validation

Figure 1.1: A system for doing knowledge discovery in gene expression data: The core activity in the system is knowledge discovery. Gene expression data is piped into a clustering module. The output from this module is the same data set now divided into groups (or clusters) of similar objects (in this case genes). This output can in turn be piped into a model module that outputs some kind of representation of the gene-cluster function. In this way we do unsupervised learning on the unlabelled data and then supervised learning on the resulting labelled data. The whole process is supported by easy access to existing domain knowledge. Hence, the results from the knowledge discovery activity can constantly be validated against already known facts. In addition, the results from the knowledge discovery activity can be evaluated by the user. Results from clustering can be visualised using some kind of graph or diagram. Results from modelling can be evaluated against some kind of performance measure telling the user how good the model is at classifying unseen cases.

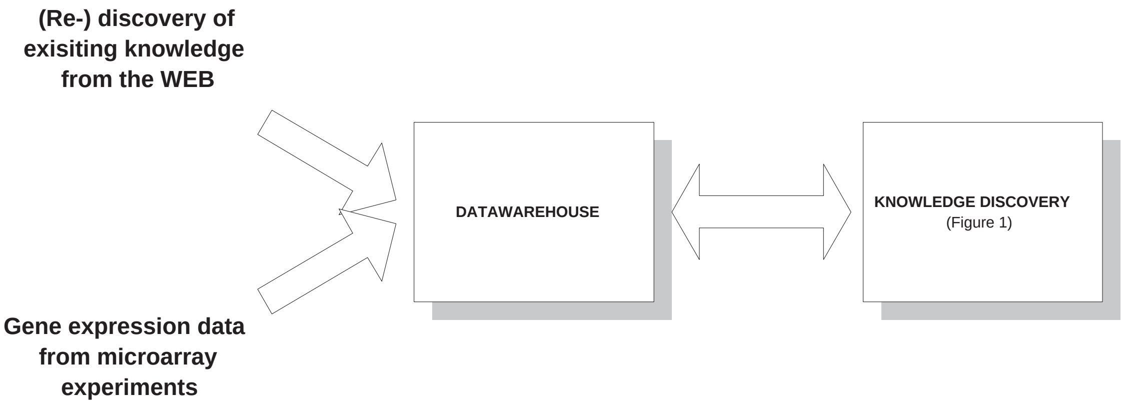

(Re-) discovery of exisiting knowledge from the WEB
Gene expression data from microarray experiments
DATAWAREHOUSE
KNOWLEDGE DISCOVERY
(Figure 1)

Figure 1.2: The context of functional genomics: Data and knowledge are stored in a data warehouse to back up the KDD process. Data are mainly collected from microarray experiments while knowledge is collected from the WEB and from the KDD process itself.

This thesis describes a large context in which several contributions have been made by a number of researchers. The specific technical contributions made by the author includes implementation of the methodology and execution of the experiments.

1.1 Reader's guide

This thesis is divided into three parts. The first part defines the problems that we want to address both from a biological and from a computer science point of view. The second part describes methods and tools designed to solve these problems, while part three contains some case studies where the methods described in the second part are used on real world data sets to solve problems described in the first part. The thesis is brought to a close with discussions, conclusions and indications about future work.

The work was done at the department of computer and information science and accordingly the thesis has been written under the assumption that the reader is familiar with some basic concepts from computer science. An effort has been made to provide as much biological knowledge deemed necessary for a computer scientist to understand the thesis. A biologist with some basic knowledge in computing should also be able to follow the thesis.

Part I

Background - Molecular Biology and Knowledge Discovery

We address the problems of extracting useful knowledge from gene expression data using computational methods. In order to do this we need to understand both the biological and the computer science aspects of this problem. This part attempts to provide that understanding.

Chapter 2

Molecular Genetics and DNA Technology

Molecular genetics is define by [Strachan and Read, 1999] as primarily being *concerned with the interrelationship between the information macromolecules DNA (deoxyribonucleic acid) and RNA (ribonucleic acid) and how these molecules are used to synthesise polypeptides, the basic components of all proteins*. Proteins play important roles in many cellular functions as enzymes, receptors, storage proteins, transport proteins, transcription factors, signalling molecules, hormones, etc. Since cells drive the development of the whole organism, it is needless to say how important it is to understand this relationship in order to gain insight into how the human organism functions.

This chapter will give a short introduction to molecular genetics and gene technology. For more detailed descriptions the reader should consult [Strachan and Read, 1999] or [Sjøberg, 1998].

2.1 DNA Structure

The DNA molecules are found in the chromosomes of the nucleus and in mitochondria in all eukaryote cells. They consist of a linear backbone of alternating sugar and phosphate residues, where the sugar, a 5 carbon sugar called deoxyribose, is successively linked by covalent phosphodiester bonds. Covalently attached to the each sugar residue is a nitrogenous base, which is either adenine (A), cytosine (C), guanine (G) or thymine (T). These bases are heterocyclic rings of carbon and nitrogen atoms. A sugar with an attached base is called a nucleoside. A nucleoside with an attached phosphate group is called a nucleotide and is the basic repeat unit of a DNA strand. RNA has a similar structure to DNA except they contain ribose sugar residues in place of deoxyribose and uracil (U) instead of thymine (T). A protein is made up of one or more polypeptide molecules. Like DNA and RNA it is a linear sequences of repeating units (in this case these units are amino acids).

In DNA molecules, each phosphate group links carbon atom 3’ of a sugar to carbon atom 5’ of the neighbouring sugar. The structure of DNA is a double helix in which two DNA molecules (DNA strands) are held together by weak hydrogen bonds. Hydrogen bonding occurs between laterally opposed base pairs of the two strands of the DNA duplex:

adenine (A) binds to thymine (T) and cytosine (C) binds to guanine (G). One end of each DNA strand will have a terminal sugar residue in which carbon atom number 5' is not linked to a neighbouring sugar residue, while the other end will have a terminal sugar residue with a similar absence of a bonding at carbon atom number 3'. These ends are called the 5' end and the 3' end respectively. The 5' $\rightarrow$ 3' direction of one DNA strand in the two strands of a DNA duplex is always opposite to that of its partner (anti-parallel). The two strands of a DNA duplex are said to be complementary since the sequence of bases from one strand can be inferred from its partners. The genetic information in the DNA is encoded by these linear sequences of bases called the primary structure.

2.2 RNA Transcription and Translation

DNA specifies the synthesis of RNA and RNA specifies the synthesis of polypeptides (proteins). This DNA $\rightarrow$ RNA $\rightarrow$ polypeptide (protein) flow of genetic information has been described as the *central dogma* of molecular biology. The first step of this process is called *transcription* and occurs in the nucleus of eukaryotic cells. The second step is called *translation* and occurs in the ribosomes of cells.

Transcription is the synthesis of RNA using DNA as a template. Normally only one of the two DNA strands acts as a template for the RNA synthesis since RNA molecules normally exist as single strands only. During transcription the double-stranded DNA is unwound and the *template strand* forms a new double-stranded RNA-DNA hybrid with the growing RNA chain. The transcript has the same 5' $\rightarrow$ 3' direction and base sequence (except that U replaces T) as the opposite, non-template strand of the double helix. For this reason the non-template strand is called the *sense strand* and the template strand is called the *anti-sense strand*. The transcription is catalysed by a large enzyme called RNA polymerase. This enzyme performs all three steps of the transcription: It unwinds the DNA helix and starts transcription, it moves along the template strand and binds the RNA nucleotides together one by one, and it terminates the transcription at the right place. There are three different main categories of RNA that are all involved in transferring the genetic information in the protein synthesis. The *messenger RNA (mRNA)* is a transcript of the sequence of bases in one part of the DNA and acts as a recipe in the protein synthesis. The *ribosomal RNA (rRNA)* helps building the ribosomes in which the protein synthesis takes place. The *transfer RNA (tRNA)* brings the correct amino acid to the correct location during the protein synthesis.

Only a small proportion of all DNA in cells is ever transcribed. Different cells transcribe different segments of the DNA according to their needs. However, only a very small portion of all cellular DNA is ever transcribed in any cell. Moreover, only a portion of the transcribed RNA is translated into polypeptides. Not all RNA molecules are mRNA and thus do not specify polypeptides directly. Also the transcribed RNA that does specify polypeptides directly is subject to processing events which discards much of the initial RNA sequence to give a much smaller mRNA. In addition, only a central part of the mature mRNA is translated while the rest remains untranslated. The relatively small segment of DNA which is transcribed into RNA is what we call *genes*. Thus different genes code different biological functions carried out by the transcribed RNA.

A ribosome can be seen as a factory for the polypeptide (or the protein) synthesis. Since the mRNA contains the recipe, the ribosome needs to bind to these mRNA strands in order to make polypeptides. When attached to the mRNA the ribosome attracts tRNA with amino acids. These amino acids bind and form polypeptides.

2.3 Regulation of Gene Expression

A gene that is transcribed and used in protein synthesis is said to be expressed in the cell. Of course not all of the around 100 000 genes in the human cells are expressed constantly. Humans have over 200 different types of cells and tissue and only a few genes are expressed in all cells. Most genes are only expressed in particular cells. Some genes are expressed only in a short time while others are more or less constantly expressed. A lot of genes are turned on and off according to the surrounding environment. An organism's ability to regulate the expression level of genes is crucial. Cancer is often the result when these control mechanisms are not working. Also the synthesis of proteins takes a lot of energy. In order to save energy the cells are totally dependent on only making proteins that are strictly needed.

An expression level of a gene can be regulated in different ways and at different levels:

1. At transcription level: Only genes that are needed are being transcribed.
2. At mRNA level: A mRNA molecule only exists for a limited amount of time. The shorter this time is the less protein is being produced.
3. At translation level: Although a gene is transcribed its mRNA needs not be translated into a protein.
4. At protein level: A protein needs not be active. It is known that, for example, enzymes can be turn on and of.

When a protein (gene product) is needed to turn a gene (the transcription) on it is called positive gene regulation. When a protein is needed to turn a gene off it is called negative gene regulation.

2.4 Fundamentals of DNA Technology

The fundamentals of DNA technology are largely based on two quite different approaches to studying specific DNA sequences within a complex DNA population. In *DNA cloning* the desired DNA fragments are selectively amplified so that it is purified essentially to homogeneity. In *molecular hybridisation* the desired DNA fragments are specifically detected within a complex mixture of many different sequences.

DNA cloning can be done in terms of *cell-based DNA cloning* or in terms of *cell-free DNA cloning*. During cell division the chromosomes in the nucleus are replicated, and this is utilised in cell-based DNA cloning. The DNA fragment of interest is attached to a DNA sequence which is capable of independent replication. The recombinant DNA fragment is then transferred into a suitable host cell where it can be propagated selectively. Cell-free DNA cloning or polymerase chain reaction (PCR) is a newer form of DNA cloning which is faster, more sensitive and more robust than cell-based cloning. PCR can be done in laboratories, outside of cells, and is based on alternating denaturation (dissociation of complimentary strands to give single-stranded DNA) and hybridisation (the situation where two complementary single-stranded DNA molecules form one double-stranded DNA molecule) of the desired DNA fragment in a mixture of all the needed basic components.

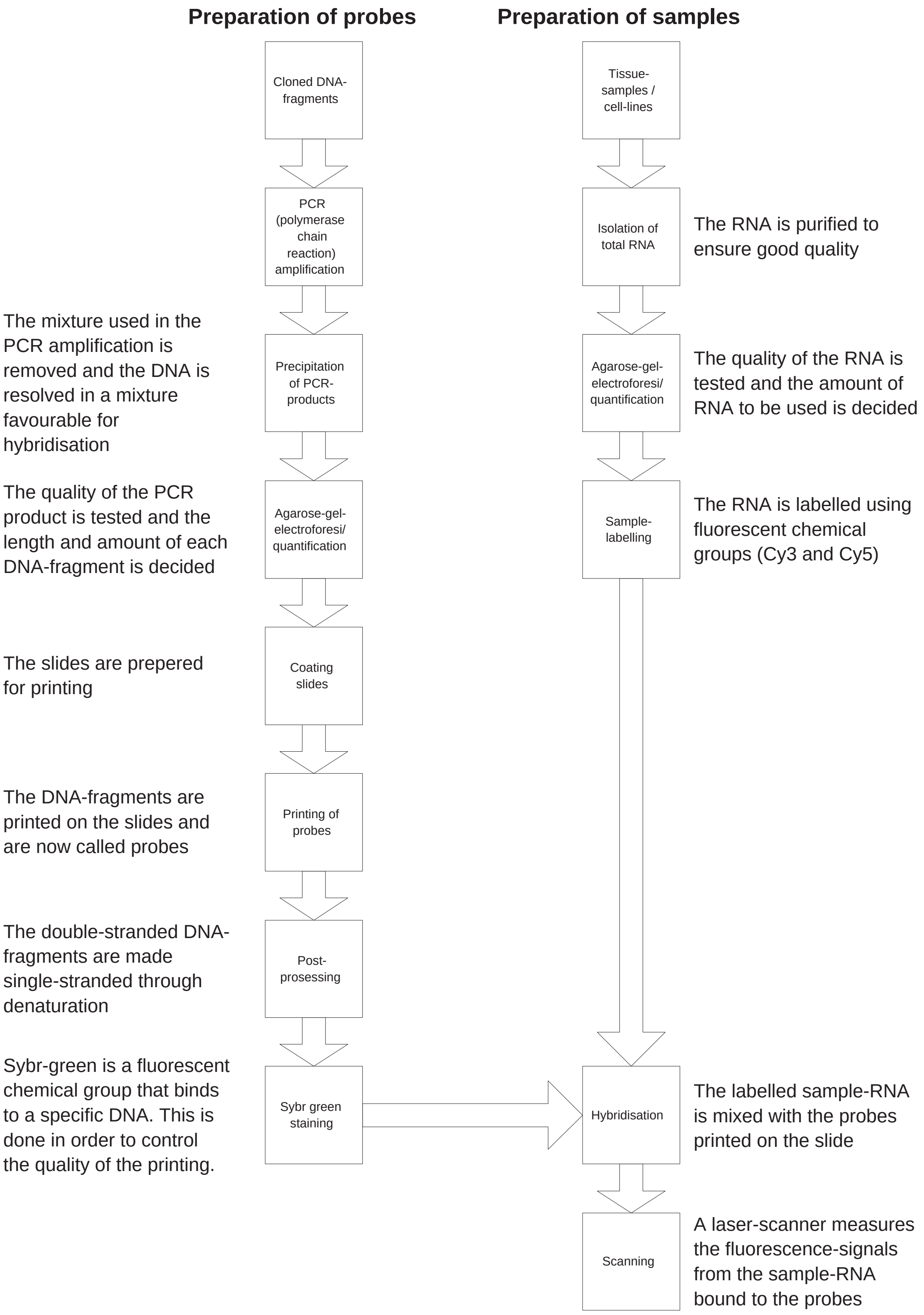
Standard DNA hybridisation assays involve using a labelled DNA probe to identify related DNA (or RNA) molecules (that is, one with a significantly high degree of sequence similarity) within a complex mixture of unlabelled DNA (or RNA) molecules. The labelled DNA probe is isolated by cell-based DNA cloning or by PCR. *DNA microarrays* provide a scale-up in hybridisation assay technology because of their huge capacity and automation. The probes consist of unlabelled DNA printed into the surface of a microscope slide. The target is labelled and in solution. The annealing of a probe DNA strand and a complementary target DNA strand indicates that the probe-sequence exists in the target. Thus the method is to use the identified probes to query the target DNA to identify fragments in the complex target which is related in sequence to the probe.

DNA microarrays have proven to be an effective method for gene expression quantification. The amount of mRNA in a cell reflects in which degree its respective gene is expressed. DNA microarrays can be used to identify which mRNA that is expressed in a given solution and in which *degree* it is expressed. Equipment for doing microarray experiments are installed at our university and this thesis is partly a result of the urgent need for computational methods and tools to analysis data from these experiments. Figure 2.1 illustrates the steps in a microarray experiment, while Figure 2.2 shows how the up and down regulation of genes is represented afterwards.

[Schena, 1999] provides a detailed description of microarrays and microarray experiments.

2.5 Computational Biology

The large amounts of data collected from DNA microarrays experiments have made the biologists aware of the importance of computational methods and tools for data analysis. The use of computer science in biology, and in particular in the analysis of gene data, is frequently referred to as *computational biology*. In the next chapter we will look at some of the challenges in computer science related to the analysis of gene expression data collected from DNA microarray experiments.



The mixture used in the PCR amplification is removed and the DNA is resolved in a mixture favourable for hybridisation

The quality of the PCR product is tested and the length and amount of each DNA-fragment is decided

The slides are prepered for printing

The DNA-fragments are printed on the slides and are now called probes

The double-stranded DNA-fragments are made single-stranded through denaturation

Sybr-green is a fluorescent chemical group that binds to a specific DNA. This is done in order to control the quality of the printing.

The RNA is purified to ensure good quality

The quality of the RNA is tested and the amount of RNA to be used is decided

The RNA is labelled using fluorescent chemical groups (Cy3 and Cy5)

The labelled sample-RNA is mixed with the probes printed on the slide

A laser-scanner measures the fluorescence-signals from the sample-RNA bound to the probes

Figure 2.1: The steps in a DNA microarray experiment. The diagram is provided by Kristin Nørsett.

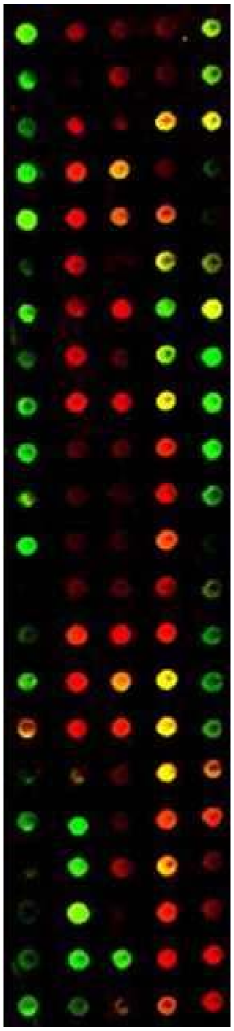


Figure 2.2: The image shows a typical result from sample-RNA being hybridised to the microarray. According to some reference it shows genes whose mRNAs are suppressed as green spots and genes whose mRNAs are expressed as red spots. Yellow spots represent genes whose expression do not change.

Chapter 3

Knowledge Discovery in Gene Expression Databases

3.1 Knowledge Discovery

Knowledge Discovery in Databases (KDD) has been defined by [Fayyad et al., 1996] as *the non-trivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data.* The KDD process has a highly interdisciplinary nature having evolved from fields like machine learning, pattern recognition, databases, statistics, AI, knowledge acquisition for expert systems, data visualisation and high-performance computing. A related field is *data warehousing* which refers to the popular business trend of collecting and cleaning data to make them available for analysis and decision support. In many ways we can say that data warehouses set the stage for KDD. KDD is really a process and involves the following basic steps:

1. Identifying the goal of the KDD process.
2. Creating a target data set.
3. Data cleaning and preprocessing.
4. Data reduction and projection.
5. Matching the goals of the KDD process (step 1) to a particular data mining method.
6. Choosing the data mining algorithm(s) and selecting method(s) to be used for searching for data patterns.
7. Data mining.
8. Interpreting the mined patterns.
9. Acting on the discovered knowledge.

Data mining is a step in the KDD process that consists of applying data analysis and discovery algorithms that produce a particular enumeration of patterns (or models) over the data. The goals of the KDD process are very much decisive for which method to use.

They can either be *verification* of some hypothesis or *discovery* of new patterns. Discovery can further be subdivided into *prediction*, that is, predicting the future value of unseen cases, or *description*, that is, finding patterns for presentation in a human-understandable form. Data mining involves the task of fitting models to observed data. This requires a model representation language, a model evaluation criterion and a search method to find the optimal model with the optimal parameters. Classically either statistical or logical formalisms are used for representing models. Most data mining methods can be classified in one of the following categories:

- Classification: learning a function that maps a data item into one of several predefined classes.
- Regression: learning a function that maps a data item to a real-value prediction variable.
- Clustering: identifying a finite set of categories or clusters to describe the data.
- Summarisation: finding a compact description for the data.
- Dependency modelling: finding a model that describes significant dependencies between variables.
- Change and deviation detection: discovering the most significant changes in the data from previously measured or normative values.

One should notice that many of the above mentioned methods are based on heuristic approximations because of the expenses of searching for optimal solutions.

Some of the most interesting challenges within KDD today includes large databases with high dimensions, overfitting when learning models, assessing of statistical significance, changing data and knowledge, missing and noisy data, complex relationships between data fields, understandability of patterns, user interaction and prior knowledge, and integration with other systems.

3.2 Knowledge Discovery in Gene Expression Databases

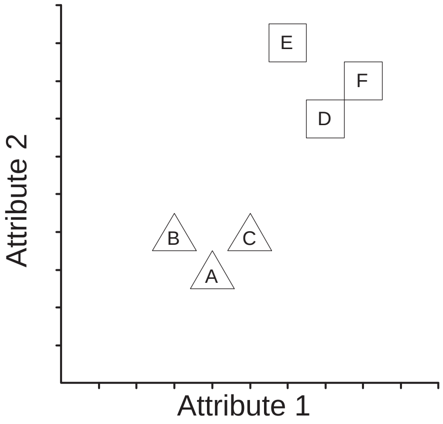
Analysing gene expression data is primarily a clustering task, although modelling can also be used as explained in Chapter 1. We want to identify a finite set of natural groups or clusters that describes the data. We then want to present the clusters with all available domain (biological) knowledge in order to make the job of interpretation as easy as possible for the domain expert (biologist). Alternatively we could compare the obtained clusters with domain knowledge automatically. This, however, requires that the domain knowledge is strictly formalised, something which is never easy to do when faced with human knowledge which tends to be both subjective, partly inconsistent and almost always incomplete.

With reference to the above discussion we will dived the task of knowledge discovery in gene expression databases into three main steps; *clustering*, *evaluation* and *validation* (see also Figure 1.1). The actual clustering algorithms will be thoroughly discussed later. This chapter is only meant to be an introduction to the problem. By evaluation we mean the task of measuring the correctness or the goodness of the clusters. Of course, this strongly depends on what we mean by a ”natural grouping”. In our case this normally boils down to finding groups of genes with similar expression patterns. Evaluation is most usually done through visualisation, although this is a very inexact measure which also requires manual inspection. There exist better metrics and these will be discussed later. For now, consider the following simple example.

EXAMPLE 3.2.1 (CLUSTERING AND EVALUATION THROUGH VISUALISATION)
Consider the table below to be a data set resulting from a microarray experiment (these data sets tend to consist of thousands of genes and tens of attributes, but as an example this data set is large enough).

Gene	*Attribute 1*	*Attribute 2*
A	*1*	*0.75*
B	*0.75*	*1*
C	*1.25*	*1*
D	*1.75*	*1.75*
E	*1.5*	*2.25*
F	*2*	*2*

Given a partitioning $C_1 = \{A, B, C\}$ and $C_2 = \{D, E, F\}$ of the data set (using some clustering algorithm), we would like to visually evaluate the clusters goodness. Since this is a two-dimensional case the data set can easily be represented in a diagram.



The conclusion in this case seems to be that the obtained clusters match our intuition of a natural grouping well. However, in many cases, the natural grouping does not come as easy as in this example and thus we have to make some kind of pragmatic decision in order to arrive at a grouping that is good enough. □

The genes in a gene expression data set are normally categorised into known genes and unknown genes. Loosely we define “known genes” to be genes with a known function. Of course, the function of a gene is to code RNA (mRNA, tRNA or rRNA) which in turn is involved in the synthesis of proteins. What we really mean about the function of a gene is the function of the protein that this gene codes. To make it even more complicated, a protein’s function can be carried out in different biological processes. A biological process is a series of events that may involve more than one cell. In this thesis we will frequently refer to the search of the function of unknown genes as the ultimate goal of knowledge discovery in gene expression databases. However, equally important is the task of describing the known genes with respect to process in the specific biological setting that we are studying.

In order to interpret computer generated results we need to validate our clusters against biological knowledge. Biological knowledge can be organised in an *ontology*. An ontology is a tree where the leaf nodes are known genes and the parent nodes are clusters of genes with similar functions. Thus an ontology is a formalised piece of domain knowledge, in this case the function of a set of genes. Given a set of clusters resulting from applying some kind of clustering algorithm to a set of gene expression data, we now want to verify these clusters against the domain knowledge contained in the ontology. Typically some of the genes we have clustered are unknown and thus do not exist in the ontology. Hopefully there is a significant overlap between our clusters and the clusters existing in the ontology. What we are looking for are unknown genes that appear in clusters with significant overlap with clusters in the ontology. Inductively we might conclude that these unknown genes are connected to the other known genes in the cluster (for example by having the same function). This is because the overlap with the cluster in the ontology indicates that this cluster is biologically relevant. The inductive assumption that the function of unknown genes can be inferred from their similarity to known genes most be used with caution. The similarity could be due to measurement errors or could simply be a coincident. If all the other clusters more or less overlap with a cluster in the ontology that definitely strengthens the hypothesis. If the biologists find the result interesting they might want to investigate it more thoroughly.

The correlation between similarity in gene expression profiles and the similarity in gene function has been discussed in various articles. Two examples are [Eisen et al., 1998] and [Pellegrini et al., 1999].

EXAMPLE 3.2.2 (CLUSTERING AND VERIFICATION)
Figure 3.1 shows an example of a clustered set of gene expression data and a corresponding ontology. This ontology describes genes involved in the different stages of the cell cycle. Note that gene N and O are not in the analysed set of gene expression data. Also note that gene R and S are unknown genes and thus do note exist in the ontology. Clusters C1 and C4 both match clusters G4 and G0 respectively while gene G seems to have been misplaced in cluster C2 instead of cluster C3. Gene S has the properties we are looking for. This gene is unknown and are clustered together with three genes all belonging to the same cluster in the ontology. This might be an interesting case for the biologist to study further? □

3.3 Related Research

In the following section we will look at some examples of published work done in the field of gene expression data analysis.

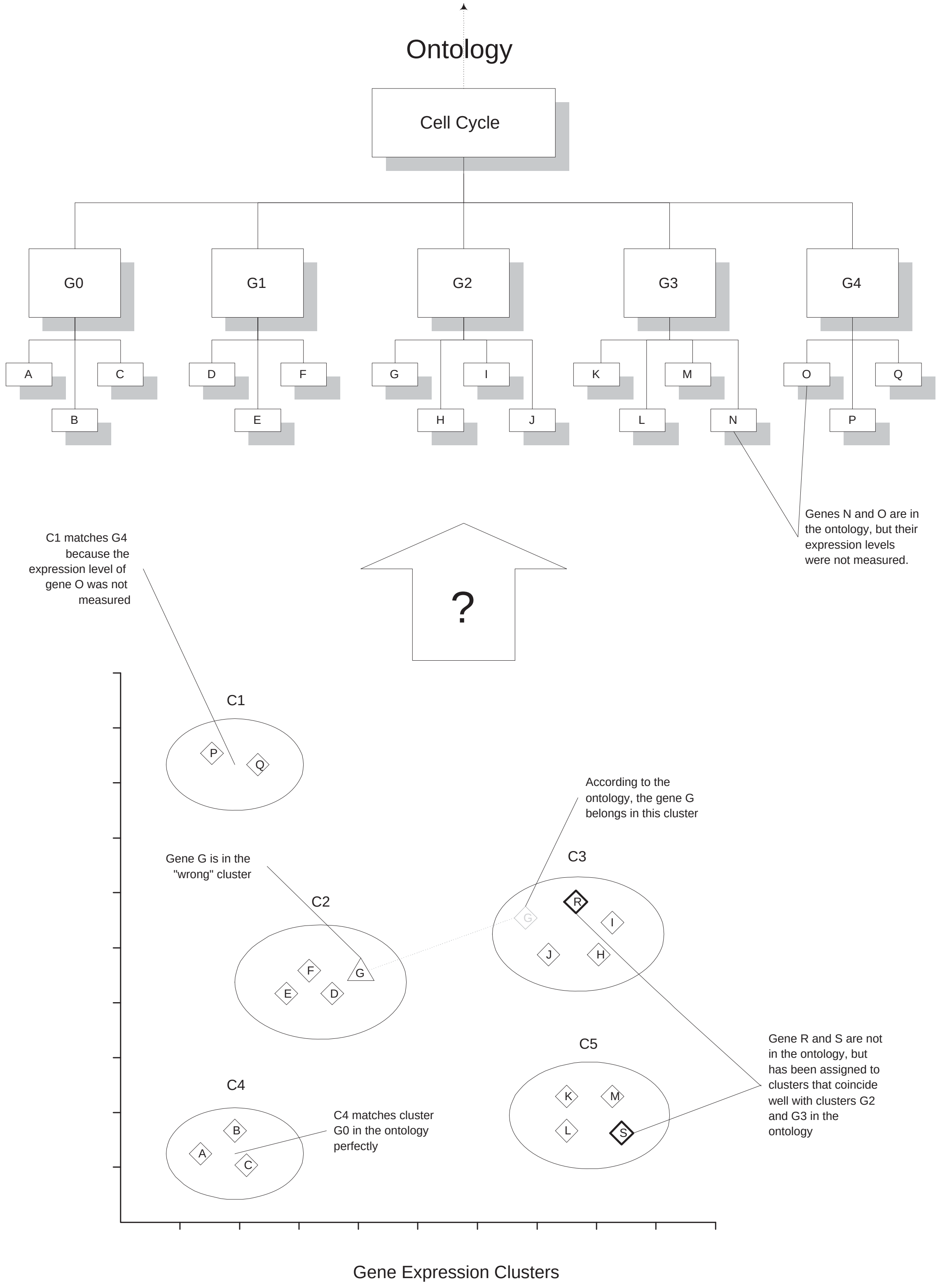

Ontology
Cell Cycle
G0
G1
G2
G3
G4
A
C
B
D
F
E
G
I
H
J
K
M
L
N
O
Q
P
Genes N and O are in the ontology, but their expression levels were not measured.
C1 matches G4 because the expression level of gene O was not measured
?
C1
P
Q
According to the ontology, the gene G belongs in this cluster
Gene G is in the "wrong" cluster
C3
R
I
G
J
H
C2
F
G
E
D
C5
Gene R and S are not in the ontology, but has been assigned to clusters that coincide well with clusters G2 and G3 in the ontology
C4
K
M
L
S
B
A
C
C4 matches cluster G0 in the ontology perfectly
Gene Expression Clusters

Figure 3.1: The process of matching a set of clusters to formalised domain knowledge (an ontology). The Figure is taken from [Tjeldvoll, 1999].

3.3.1 The Transcriptional Program in the Response of Human Fibroblast to Serum

This article ([Iyer et al., 1999]) studies the human fibroblasts response to serum which appears to be related to the physiology of wound repair. The temporal changes in mRNA level of 8613 human genes were measured at 12 times ranging from 0 minutes to 24 hours after serum stimulation. A subset of 517 genes whose expression changed substantially in response to serum was selected for further analysis (this data set is publicly available on the WEB: $<$ http : //genome − www.standford.edu/serum $>$).

The subset of 517 genes was clustered hierarchical into groups on the basis of the similarity of genes expression profiles over the full 24 hours. Ten clusters were identified containing 452 of the 517 genes. These cluster were presented by their average expression profiles and a biological interpretation was given.

Measurements done at different time points results in a set of values for each gene called a *time series*. Unlike normal measurements which represent a point in the n-dimensional space, time series can easily be represented as a function of time in two-dimensions. Also time series analysis differs from other data analysis in that we are looking for similarity in terms of curve-similarity. This often requires some kind of transformation of the data into gradients or even components using mathematical transformations.

The ”fibroblast data” data set will be thoroughly analysed and discussed in Part III.

3.3.2 Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays

This article ([Alon et al., 1999]) investigates the expression profile of over 6500 genes in 40 tumour and 22 normal colon tissue samples (this data set is publicly available on the WEB: $<$ http : //www.molbio.princeton.edu/colondata $>$). A subset of 2000 genes with the highest minimal intensity across the samples was analysed.

The data was clustered both with respect to genes and with respect to tissue. The method used was the one of organising the data sets into a binary tree where genes are near each other one the ”gene tree” if they show a strong correlation across experiments, and tissue are near each other on the ”tissue tree” if they have similar gene expression profiles. In this way they were able to classify genes into functional groups and to classify tissue based one gene expression.

Note that in this example the data already has a known classification; tumour tissue and normal tissue. Thus we can use another data mining method, classification, where we learn the function that maps the gene expressions into one of these predefined classes (tumour tissue or normal tissue). This could be useful when looking for genes with predictive capabilities.

The “tumour and normal colon tissues” data set will also be analysed in Part III.

3.3.3 Large-scale temporal gene expression mapping of central nervous system development

This article ([Wen et al., 1998]) investigates the expression profile of 112 genes in the rat central nervous system and is another example of time series data sets. The data set is clustered on the basis of similarity in the time series' slopes and shows that the functional classes clearly map to particular expression profiles.

3.4 Our Task

From a computer science point of view knowledge discovery in gene expression databases includes many of the classical challenges in KDD listed earlier. We are faced with large databases that most probably contain noise. We are faced with knowledge that is constantly changing as a result of new discoveries. And we are also faced with the problem of integrating domain knowledge into the KDD process. In order to summarise this chapter we will go through the steps of the KDD process specifically with our task in mind: Our goal for the KDD process will be to discover, or at least set the stage for a biologist to discover, functions of unknown genes. We will analyse gene expression data sets collected from microarray experiments. The obvious data mining tool for this task is clustering algorithms together with different data preprocessing method very much dependent one what kind of data we are going to analyse. Alternatively, we might want to use modelling method instead of clustering methods or as a supplement to clustering methods. The minded clusters will be interpreted by biological experts in the light of existing domain knowledge in order to give them a bio-medical value.

Part II

Cluster Analysis

Unsupervised learning is the task of learning without feedback. This is called clustering, because the objective is to reveal the underlying natural groups in a set of example data. In this part we will examine different methods for this purpose, including hierarchical clustering, k-means, indiscernibility-based clustering, self organising maps, etc. Some of these methods will later be used in case studies on real-world gene expression data.

Chapter 4

Knowledge and Learning

Artificial intelligence (AI) was formally initiated as a sub-field of computer science in 1956. However, the definition of intelligence is still not agreed upon. According to [Russel and Norvig, 1995] definitions vary along two dimensions; whether we measure success in terms of human performance or in terms of some ideal concept on one hand, and whether we are concerned with reasoning or behaviour on the other hand. The classical definition proposed by Alan Turing in 1950 measures intelligence against human behaviour. The Turing Test states that a computer system is intelligent if a human interrogator can not tell weather it is communicating with a computer or a human being. Modern approaches to AI are mainly concerned with building entities that act rationally, that is, do the right thing given their existing knowledge and present input. After all, every problem can be formulated as a search problem where the goal is to find a sequence of actions that take the system from its present state to a predefined goal state. With this in mind there is no reason to imitate humans in order to build intelligent entities. That is of course if the primary interest is not at constructing theories about the working of the human mind itself (cognitive science). Either-way the Turing Test raises interesting problems that are research areas in AI today. These include natural language processing, knowledge representation, automated reasoning and machine learning.

Knowledge has a central role in AI as it seems clear that intelligence requires knowledge. Critical to the success of an intelligent entity is its ability to reason with existing, possibly uncertain, knowledge, to adopt new knowledge from learning and to represent the knowledge in a way that makes reasoning possible. Our primary goal is to build entities that extract knowledge from (possibly) large data sets. This requires theories for learning from examples and for representing extracted knowledge in an appropriate way. Furthermore it requires the ability to incorporate existing knowledge, often called expert knowledge or domain knowledge, to help the KDD process. In the following sections we will look more deeply into knowledge and learning in order to define and understand these critical aspects of knowledge discovery.

General introduction to AI and further discussions on ”What is artificial intelligence?” can be found in e.g. [Russel and Norvig, 1995] and [Nilsson, 1998].

4.1 Knowledge

In the same way as intelligence has different and partly conflicting definitions so has knowledge. Four decades of AI have however taught us that knowledge possesses some less desirable properties such as being voluminous, hard to characterise and constantly changing. Also there has been pointed out many ways of categorising knowledge types; induced knowledge vs. deduced knowledge, causal knowledge vs. diagnostic knowledge, crisp knowledge vs. uncertain knowledge, quantitative knowledge vs. qualitative (or common-sense) knowledge, etc.

In the context of knowledge discovery in databases it is interesting to see knowledge in connection with data. Knowledge differs from data in that it is organised. In fact, one can define knowledge as the ability to organise, or classify, data. This approach was taken by [Pawlak, 1991] and will be adopted here. Pawlak argued that knowledge is a partition of a *universe*, that is, some real or abstract world represented by a finite set of *objects*. Consider for example the ability to pick the right wine for the right food. The knowledge required is the ability to partition the universe of all wines into classes labelled with correct type of food. Or consider a more complex task as driving a car. Again the knowledge required is the ability to partition the universe of all situations into classes labelled with the correct action (”hit the breaks”, ”turn left”, etc).

We already mentioned that knowledge representation is a crucial factor in all intelligent entities. The object of knowledge representation is to express knowledge in computer-tractable form. We will adopt the knowledge representation language proposed by Pawlak in rough set theory ([Pawlak, 1982], [Komorowski et al., 2000a] and [Nguyen and Nguyen, 1996]). A data set is represented in a table such that each row represents objects and each column represents some property that can be measured for each object.

Definition 4.1.1 (Information system)
An information system is a pair $\mathcal{A} = (U, A)$ where U is a non-empty finite set of objects called the universe and A is a non-empty finite set of attributes such that $a : U \rightarrow V_A$ for every $a \in A$. The set V_a is called the value set of a. $\Box$

Knowledge contained in an information system can be expressed by an equivalence relation. An equivalence relation is a binary relation $R \subseteq X \times X$ where X is the *equivalence class* of $x \in X$ if all objects $y \in X$ is such that xRy. Any equivalence relation is defined to be reflexive (xRx), symmetric (if xRy then yRx) and transitive (if xRy and yRz then xRz). Given an information system $\mathcal{A} = (U, A)$ there is always associated a equivalence relation $IND_{\mathcal{A}}(B)$ with every $B \subseteq A$.

$$IND_{\mathcal{A}}(B) = \{(x, y) \in U^2 \mid \forall a \in B\ a(x) = a(y)\} \tag{4.1}$$

$IND_{\mathcal{A}}(B)$ is called the *B-indiscernibility relation* and its classes are denoted $[x]_B$.

The available knowledge in an information system is often called the *knowledge base* and can be defined as a pair $K = (U, R)$. Notice that the partition determined by the equivalence relation R gives us atomic sets, that is, objects in these sets are similar with respect to the given attributes and thus can not be discerned. Consequently, knowledge needs to be defined approximately. Let $\mathcal{A} = (U, A)$ be an information system and let $B \subseteq A$ and $X \subseteq U$. The set X can now be approximated using only the information in B by constructing the *B-lower* and *B-upper approximations of X*. These approximations are denoted $\underline{B}X$ and $\overline{B}X$ respectively and are defined by $\underline{B}X = \{x|[x]_B \subseteq X\}$ and $\overline{B}X = \{x|[x]_B \cap X \neq \emptyset\}$. The set X is said to be *rough* if $\overline{B}X - \underline{B}X$ is non-empty and *crisp* otherwise.

	Wine district	Main grape variety	Vintage	Storage temp.
x_1	Bordeaux	Cabernet Sauvignon	1992	12-15
x_2	Rhône	Syrah	1992	<12
x_3	Chile	Cabernet Sauvignon	1995	12-15
x_4	Bordeaux	Merlot	1995	>15
x_5	Chile	Cabernet Sauvignon	1995	12-15
x_6	Rhône	Merlot	1992	12-15
x_7	Bordeaux	Merlot	1995	>15
x_8	Chile	Merlot	1992	<12

Table 4.1: The information system $\mathcal{A}_{\text{Example}} = (\{x_1, x_2, ..., x_8\}, \{\text{Wine district}, \text{Main grape variety}, \text{Vintage}, \text{Storage temp.}\})$

EXAMPLE 4.1.1 (INFORMATION SYSTEM, KNOWLEDGE BASE, ROUGH SET)
Table 4.1 is an example of a information system that contains information about eight different wines. Some labelled example-partitions are given here:

$$IND_{\mathcal{A}_{\mathit{Example}}}(\{\textit{Wine district}\}) = \\ \{\{x_1, x_4, x_7\}_{\mathit{Bordeaux}}, \{x_2, x_6\}_{\mathit{Rhône}}, \{x_3, x_5, x_8\}_{\mathit{Chile}}\} \tag{4.2}$$

$$IND_{\mathcal{A}_{\mathit{Example}}}(\{\textit{Main grape variety}\}) = \\ \{\{x_1, x_3, x_5\}_{\mathit{Cabernet\ Sauvignon}}, \{x_2\}_{\mathit{Syrah}}, \{x_4, x_6, x_7, x_8\}_{\mathit{Merlot}}\} \tag{4.3}$$

$$IND_{\mathcal{A}_{\mathit{Example}}}(\{\textit{Vintage}\}) = \{\{x_1, x_2, x_6, x_8\}_{1992}, \{x_3, x_4, x_5, x_7\}_{1995}\} \tag{4.4}$$

$$IND_{\mathcal{A}_{\mathit{Example}}}(\{\textit{Storage temp.}\}) = \{\{x_1, x_3, x_5, x_6\}_{12-15}, \{x_2, x_8\}_{<12}, \{x_4, x_7\}_{>15}\} \tag{4.5}$$

Using all the attributes results in the following partition:

$$IND_{\mathcal{A}_{\mathit{Example}}}(A_{\mathit{Example}}) = \{\{x_1\}, \{x_2\}, \{x_3, x_5\}, \{x_4, x_7\}, \{x_6\}, \{x_8\}\} \tag{4.6}$$

The available knowledge in the system can be expressed as the knowledge base $K_{\mathit{Example}} = (U_{\mathit{Example}}, R_{\mathit{Example}})$, where R_{Example} is the equivalence relation $IND_{\mathcal{A}_{\mathit{Example}}}(B)$ and B is every subset of A_{Example}.

Now consider for example the set $X = \{x_3, x_4, x_8\}$. This set can be approximated using the definitions of rough sets:

$$\underline{A}_{\mathit{Example}}X = \{x_8\} \tag{4.7}$$

$$\overline{A}_{\mathit{Example}}X = \{x_3, x_5, x_4, x_7, x_8\} \tag{4.8}$$

From Equation 4.6 we see that x_3 is indiscernible from x_5 and that x_4 is indiscernible from x_7. Thus the set X can not be crisply defined given the available data. □

Often available is classificatory knowledge provided by an expert. This knowledge can be added as an extra attribute or label to each object.

DEFINITION 4.1.2 (DECISION SYSTEM)
A decision system is any information system of the form $\mathcal{A} = (U, A \cup \{d\})$, where $d \notin A$ is the decision attribute. The elements of A are called conditional attributes or simply conditions. The cardinality of the image $d(U) = \{k \mid d(x) = k, x \in U\}$ is called the rank of d and is denoted $r(d)$. □

The decision d determines a partition $\{X^1_{\mathcal{A}}, X^2_{\mathcal{A}}, ..., X^{r(d)}_{\mathcal{A}}\}$ where $X^k_{\mathcal{A}} = \{x \in U \mid d(x) = v^k_d\}$ for $1 \leq k \leq r(d)$. Hence we now have a representation of the knowledge first introduced by the expert through the decision d and now retained in the decision table. However, this knowledge is itself not as powerful as the knowledge possessed by the expert in that it does not reflect any general classificatory capability. Most often the decision system contains only a small portion of all thinkable objects. Thus what we want is the knowledge needed to be able to do the classification, not the knowledge of the classes themselves. The solution to this problem is to view the available decision table as a set of training examples. The underlying general knowledge can by learned from these examples and the learned knowledge can in turn be used to classify unseen objects. Learning will be discussed in the next section.

Given a decision system $\mathcal{A} = (U, A \cup \{d\})$ and the indiscernibility relation $IND_{\mathcal{A}}(B)$ we define the *generalised decision in $\mathcal{A}$* to be the function $\delta_A : U \rightarrow \mathcal{P}(V_d)$ defined by $\delta_A(x) = \{i \mid \exists y \in U \; y \; IND_{\mathcal{A}}(A) \; x$ and $d(y) = i\}$. If $|\delta_A(x)| > 1$ for any $x \in U$ the decision system is said to be *inconsistent (non-deterministic)*, that is, there exist objects that are indiscernible but belong to different decision classes. Thus the decision classes themselves are rough sets and consequently our knowledge given the available data is approximate. Notice that this will always be the case to some extent since the projection from the real world to data itself is an approximation.

	Wine district	Main grape variety	Vintage	Storage temp.	Decision
x_1	Bordeaux	Cabernet Sauvignon	1992	12-15	Drink now
x_2	Rhône	Syrah	1992	<12	Hold
x_3	Chile	Cabernet Sauvignon	1995	12-15	Drink now
x_4	Bordeaux	Merlot	1995	>15	Drink now
x_5	Chile	Cabernet Sauvignon	1995	12-15	Hold
x_6	Rhône	Merlot	1992	12-15	Hold
x_7	Bordeaux	Merlot	1995	>15	Drink now
x_8	Chile	Merlot	1992	<12	Hold

Table 4.2: The decision system $\mathcal{A}_{\text{Example}} = (\{x_1, x_2, ..., x_8\}, \{\text{Wine district}, \text{Main grape variety}, \text{Vintage}, \text{Storage temp.}\} \cup \{\text{Decision}\})$

EXAMPLE 4.1.2 (DECISION SYSTEM)
Table 4.2 is an example of a decision system constructed from the information system in Example 4.1.1 by adding an extra attribute. This additional attribute states whether or not the wine has been stored for a sufficient period of time and determines a partition of the wine into two labelled equivalence classes (decision classes):

$$U_{\mathit{Example}}/\mathit{Decision} = \{\{x_1, x_3, x_4, x_7\}_{\mathit{Drink\ now}}, \{x_2, x_5, x_6, x_8\}_{\mathit{Hold}}\} \tag{4.9}$$

Let the two decision classes be denoted D_{Drink} and D_{Hold}. As before these classes can be approximated using rough set theory:

$$\underline{A}_{Example}D_{Drink} = \{x_1, x_4, x_7\} \tag{4.10}$$

$$\overline{B}_{Example}D_{Drink} = \{x_1, x_3, x_5, x_4, x_7\} \tag{4.11}$$

$$\underline{A}_{Example}D_{Hold} = \{x_2, x_6, x_8\} \tag{4.12}$$

$$\overline{B}_{Example}D_{Hold} = \{x_2, x_5, x_3, x_6, x_8\} \tag{4.13}$$

Obviously the decision classes can't be crisply defined and consequently our decision system is inconsistent. □

4.2 Machine Learning

[Mitchell, 1997] proposes a broad definition of machine learning in that he includes all computer programs that improves its performance at some task through experience. In principle all learning can be seen as learning the representation of a function. Take for example the task of driving a car. The problem is to learn what action to take (”turn left”, ”hit the breaks”, etc.) given different situations. This problem can be reformulated as the problem of learning the function that maps the input (all possible situations on the road) to the correct output (actions).

Machine Learning (ML) takes different forms dependent on the degree of feedback and prior knowledge available. Any situation in which both the input and the correct output are available is called *supervised learning*. The system is being told the correct output and evaluates its proposed output against this feedback. On the other hand, *reinforcement learning* is the situation where the system receives some evaluation of it output but is not told the correct output. Learning when there is no hint at all about the correct output is called *unsupervised learning*. An unsupervised learner can only learn patterns or relationships among it own input. In addition to the three types of learning situations mentioned so far (supervised, reinforcement and unsupervised), learning can be either inductive or analytical. *Inductive learning* is the task of approximating a function f from a given set of example-pairs $(x, f(x))$. The approximation is called a hypothesis and should by consistent with all the training examples. In *analytical learning*, however, the system is provided with both training examples and a domain theory, and the approximated function should be consistent with them both. Most existing systems and methods are inductive approaches although additional prior knowledge should help a lot in most learning situations. The reason is probably the fact that the task of representing human expert knowledge in a computer system is extremely complex given the problems of knowledge representation, consistency, reasoning, etc.

A thorough coverage of the field of machine learning can be found in e.g. [Mitchell, 1997].

In this thesis we will be concerned with inductive learning, and in particular with unsupervised learning. Although domain theory (biological knowledge) will be used both for selecting the most interesting experiments and for evaluating the results, this knowledge will not take an active part in the learning task as in analytical learning. In the following we will discuss unsupervised and supervised inductive learning. We will use the notations of rough set theory as defined above and also use the rough set framework as an example of how to do unsupervised and supervised learning. In the next two chapters a more in depth coverage of unsupervised methods will be provided.

4.2.1 Unsupervised Learning

Given an information system $\mathcal{A} = (U, A)$ the task of unsupervised learning is to search for ”natural” groups of similar objects called *clusters*. Thus we want to find a partition $U = \{X^1_{\mathcal{A}}, X^2_{\mathcal{A}}, ..., X^m_{\mathcal{A}}\}$. In general this requires two things:

- A definition of similarity. What makes two objects similar? This of course is totally dependent on what we are interested in. Take for example the task of partitioning all known wines into ”natural” groups. What make two bottles of wine similar? That they are produced in the same country? That they taste the same? The last definition even requires an expert in wine tasting to do the job right. These are typical problems of defining similarity and often require close co-operation with domain experts in order to make the correct assumptions.
- An algorithm that takes an information system and a definition of similarity as input and returns a decision system with the decision classes being the obtained clusters.

EXAMPLE 4.2.1 (UNSUPERVISED LEARNING)
Consider again the information system $\mathcal{A}_{Example}$ in Table 4.1 and the following definitions:

- *Define two objects to be similar if they satisfy the indiscernibility relation $IND_{\mathcal{A}_{Example}}(A_{Example})$.*
- *Define the clusters simply to be the classes determined by the defined indiscernibility relation.*

Doing unsupervised inductive learning on the data now results in the following grouping:

$$\{x_1\}, \{x_2\}, \{x_3, x_5\}, \{x_4, x_7\}, \{x_6\}, \{x_8\} \tag{4.14}$$

Note that this approach to unsupervised learning simply results in clusters that equal the equivalence classes of the information system (Equation 4.6). □

In the next two chapters we will look more closely at the challenges of doing unsupervised learning and also indicate more advanced similarity measures and algorithms.

4.2.2 Supervised Learning

Given a decision system $\mathcal{A} = (U, A \cup \{d\})$ the task of supervised learning is to find a representation of the function that maps the set of attributes A to the decision d: $A^{|A|} \rightarrow V_d$. The rough set theory gives us a sound framework for inducing a set of propositional rules that represents this function. We will call a representation of this kind a *model*. There exist multiple other methods for modelling these kind of functions including decision trees, artificial neural networks, Bayesian belief networks, etc.

To show how rules can be induced from a decision system we need to define the concept of *reducts*. A reduct of a given decision system $\mathcal{A} = (U, A \cup \{d\})$ is a minimal set of attributes $B \subseteq A$ such that $IND_{\mathcal{A}}(B) = IND_{\mathcal{A}}(A)$. Finding the set of all minimal reducts is NP-hard ([Skowron and Rauszer, 1992]), but there exist good heuristics that compute subsets of all possible reducts in acceptable time. In principle, reducts can be found using the general scheme of *Boolean reasoning* ([Brown, 1990]):

DEFINITION 4.2.1 (BOOLEAN REASONING)
Boolean reasoning can be used to solve a problem P by applying the following general algorithm:

1. *Formulate the problem P as a system of Boolean equations.*

$$P = \begin{cases} g_1 & = h_1 \\ & \vdots \\ g_k & = h_k \end{cases} \tag{4.15}$$

   where g_i and h_i are Boolean functions $(\{0,1\}^m \rightarrow \{0,1\})$.

2. *Reduce the equation system to a single Boolean equation $f_P = 0$.*

$$f_P = \sum_{i=1}^{k} (g_i' \cdot h_i + g_i \cdot h_i') \tag{4.16}$$

   where g_i' and h_i' are the complements of g_i and h_i respectively.

3. *Compute the prime implicants[1] of f_P.*

4. *Obtain the solutions to P by interpreting the prime implicants of f_P.*

□

Given an information system $\mathcal{A}$ with n objects, the *discernibility matrix* of $\mathcal{A}$ is given by an $n \times n$ matrix with entries c_{ij}.

$$c_{ij} = \{a \in A \mid a(x_i) \neq a(x_j)\} \text{ for } i, j = 1, ..., n \tag{4.17}$$

The *decision-relative discernibility matrix* to a corresponding decision system can be constructed from Equation 4.17.

$$c_{ij}^d = \begin{cases} \varnothing & \text{if } d(x_i) = d(x_j) \\ c_{ij} - d & \text{otherwise} \end{cases} \tag{4.18}$$

[1] A *prime implicant* can be explained by the following chain of arguments: A literal is a variable or its negation. A *term* is a conjunction of literals. An *implicant* of a Boolean function f is a term p such that if the value of p is true under an arbitrary valuation v then the value of f under v is also true. A prime implicant is an implicant of f that ceases to be so if any of its literals are removed.

The *discernibility function* $f_{\mathcal{A}}$ (respectively *decision-relative discernibility function* $f_{\mathcal{A}}^{d}$) can now be defined from the discernibility matrix (respectively decision-relative discernibility matrix)

$$f_{\mathcal{A}}^{(d)}(a_1^*, ..., a_m^*) = \bigwedge \left\{ \bigvee c_{ij}^{(d)*} \mid 1 \leq j \leq i \leq n,\ c_{ij}^{(d)} \neq \varnothing \right\} \tag{4.19}$$

where $a_i^*, ..., a_m^*$ correspond to the attributes $a_1, ..., a_m$ and $c_{ij}^{(d)*} = \{a^* \mid a \in c_{ij}^{(d)}\}$.

The set of all prime implicants of $f_{\mathcal{A}}^{(d)}$ can be interpreted as the set of all reducts (respectively decision-relative reducts) of $\mathcal{A}$. Thus we have a mathematical machinery for finding reducts. From the decision-relative reducts we can now construct minimal propositional decision rules of the form $(a_1 = v_1) \wedge (a_2 = v_2) \wedge ... \wedge (a_k = v_k) \to d = v_d$ by overlaying the set of attributes in a reduct B over an object $x \in U$ and reading off the values of x for every $a \in B$ and decision d. The set of induced rules gives us a representation (or a model) of the decision system. This model has two properties. Firstly, it holds *descriptive* knowledge. It describes the most important underlying patterns and relations in the data. Secondly, it holds *predictive* knowledge. We can use the rules in the model to classify unseen objects.

The ROSETTA system is a toolkit for rough set analysis developed at our group in collaboration with Warsaw University ([Komorowski et al., 2000b], [Øhrn et al., 1998] and [Øhrn, 1999b]). A complete description of rough sets and indiscernibility-based analysis in medicine can be found in [Øhrn, 1999a]. [Vinterbo, 1999] discusses predictive models for use in medicine in general. The ROSETTA system can be used in a wide variety of different classification problem not only related to medicine. Some examples are [Tjeldvoll et al., 1999], [Hvidsten et al., 1999a] and [Øhrn and Komorowski, 1999].

In order for a model to be descriptive it needs to consist of a reasonable number of rules. Large models have no descriptive capability in practice since they are too large to be inspected by humans. According to the principle of Occam's razor, the simplest of two models both consistent with the training examples should be chosen. [Ågotnes, 1999] and [Løken, 1999] both shows how large propositional rule models can be simplified while retaining predictive performance. A shorter version of this work can be found in [Ågotnes et al., 1999]. Some of their work is even implemented in the ROSETTA system.

EXAMPLE 4.2.2 (SUPERVISED LEARNING)
Consider again the decision system in Table 4.2. Notice that {Wine district, Main grape variety}, {Wine district, Vintage}, {Main grape variety, Vintage} and {Wine district, Storage temp.} are the possible decision-relative reducts. From these reducts we can generate 22 propositional rules depicted in Table 4.3. □

As mentioned earlier, models resulting from inductive supervised learning posses predictive capabilities in the sense that they can classify unseen objects. By unseen objects we mean objects that are not a part of the training set. For this reason models are often called *classifiers*. A classifier $\hat{d}$ over a decision system $\mathcal{A} = (U, A \cup \{d\})$ is a function that maps an object $x \in U$ to a value in the value set V_d of d.

$$\hat{d}\ :\ U \to V_d \tag{4.20}$$

It should be stressed that there are actually two decision systems involved. One decision system $\mathcal{A} = (U, A \cup \{d\})$ which is the actual true system that we are trying to model

Main grape variety(Cabernet Sauvignon)	AND	Wine district(Bordeaux)	→	Decision(Drink now)
Main grape variety(Syrah)	AND	Wine district(Rhône)	→	Decision(Hold)
Main grape variety(Cabernet Sauvignon)	AND	Wine district(Chile)	→	Decision(Drink now) OR Decision(Hold)
Main grape variety(Merlot)	AND	Wine district(Bordeaux)	→	Decision(Drink now)
Main grape variety(Merlot)	AND	Wine district(Rhône)	→	Decision(Hold)
Main grape variety(Merlot)	AND	Wine district(Chile)	→	Decision(Hold)
Wine district(Bordeaux)	AND	Vintage(1992)	→	Decision(Drink now)
Wine district(Rhône)	AND	Vintage(1992)	→	Decision(Hold)
Wine district(Chile)	AND	Vintage(1995)	→	Decision(Drink now) OR Decision(Hold)
Wine district(Bordeaux)	AND	Vintage(1995)	→	Decision(Drink now)
Wine district(Chile)	AND	Vintage(1992)	→	Decision(Hold)
Wine district(Bordeaux)	AND	Storage temp.(12-15)	→	Decision(Drink now)
Wine district(Rhône)	AND	Storage temp.(<12)	→	Decision(Hold)
Wine district(Chile)	AND	Storage temp.(12-15)	→	Decision(Drink now) OR Decision(Hold)
Wine district(Bordeaux)	AND	Storage temp.(>15)	→	Decision(Drink now)
Wine district(Rhône)	AND	Storage temp.(12-15)	→	Decision(Hold)
Wine district(Chile)	AND	Storage temp.(<12)	→	Decision(Hold)
Main grape variety(Cabernet Sauvignon)	AND	Vintage(1992)	→	Decision(Drink now)
Main grape variety(Syrah)	AND	Vintage(1992)	→	Decision(Hold)
Main grape variety(Cabernet Sauvignon)	AND	Vintage(1995)	→	Decision(Drink now) OR Decision(Hold)
Main grape variety(Merlot)	AND	Vintage(1995)	→	Decision(Drink now)
Main grape variety(Merlot)	AND	Vintage(1992)	→	Decision(Hold)

Table 4.3: The rule model induced from the decision system $\mathcal{A}_{Example}$

(a subset of this system is what we call the training set), and one decision system $\mathcal{A}_{\hat{d}} = (U, A \cup \{\hat{d}\})$ which results from using the classifier $\hat{d}$ one the information system $\mathcal{A} = (U, A)$. Given an object $x \in U$ we say that $\hat{d}(x)$ is the *predicted decision of* x and that $d(x)$ is the *actual or true decision of x* [2]. Of course we want

$$(\forall x \in U)\ \hat{d}(x) = d(x) \tag{4.21}$$

to be true. Consider as a special case the situation where we have a decision system with two decision classes X_0 and X_1. The classifier is now called a *binary classifier* and takes the following form.

$$\hat{d} : U \overset{\Phi}{\to} [0,1] \overset{\theta}{\to} \{0,1\} \tag{4.22}$$

Φ is most often a *voting algorithm*. Given one object to classify, this algorithm scans the set of rules in $\hat{d}$ and assigns a number of ”votes” to each value in the value set $V_d = \{0, 1\}$. θ is a simple threshold function.

$$\Phi(x) = \frac{\text{Number of votes given to decision value 1}}{\text{Number of votes given all together}} \tag{4.23}$$

$$\theta(\Phi(x)) = \begin{cases} 1 & \text{if } \Phi(x) \geq \tau \\ 0 & \text{otherwise} \end{cases} \tag{4.24}$$

We will consider two important concepts that are used to evaluate classifiers. *Discrimination* measures how good the classifier is at guessing the correct value $d(x)$ when presented with object x. To understand this, consider a *confusion matrix C* which is a $r(d) \times r(d)$ matrix where entry $c(i, j)$ is the number of objects that really belong to class i, but were classified by $\hat{d}$ as belonging to class j.

$$\begin{array}{cc|cc} & & \multicolumn{2}{c}{\hat{d}} \\ & & 0 & 1 \\ \hline d & 0 & TN & FP \\ & 1 & FN & TP \end{array}$$

Here TN is interpreted as “true negatives”, TP as “true positives”, FN as “false negatives” and FP as “false positives”. Three important quantities for discrimination are *sensitivity*, *specificity* and *accuracy*:

$TP/(TP + FN)$	Sensitivity	$\Pr(\hat{d}(x) = 1 \mid d(x) = 1)$
$TN/(TN + FP)$	Specificity	$\Pr(\hat{d}(x) = 0 \mid d(x) = 0)$
$(TP + TN)/(TP + TN + FP + FN)$	Accuracy	

A frequently used graphical representation of discrimination is the *receiver operating characteristic* (ROC) curve. An ROC curve describes the behaviour of a classifier as the threshold τ (see Equation 4.24) is varied across the full spectrum of possible values. Each point on the ROC curve represents a different confusion matrix with different sensitivity and specificity, all defined by the given threshold value. An example of an ROC curve is given

[2] We should not forget that this value is obtained from an expert and that it too can be wrong or, as mentioned earlier, inconsistent with other values.

in Figure 4.1. It is common to interpret the *area under the ROC curve,* AUC, as the highest obtainable accuracy, given that the best threshold τ is used.

Calibration is the other concept with which we measure classifiers. It is a measure of how close the output $\Phi(x)$ of a classifier is to the probability Pr$(d(x) = 1 \mid x)$. Although discrimination is the most intuitive measure, calibration is important because human experts often want to use this value as a decision base rather than trusting the automated classification given by θ.

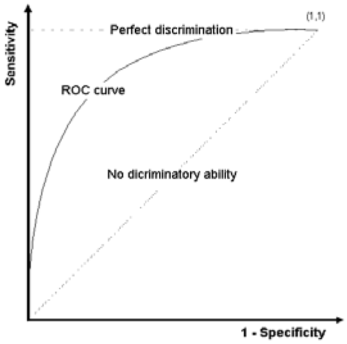


Figure 4.1: An example of an ROC curve

In order to evaluate a classifier we need objects with known decision values. Of course we can't use the objects from which we induce the model since these ”testing” objects are supposed to be unseen. Consequently the classical approach in inductive machine learning is to split the set of available example-objects into a *training set* and a *test set*. The training set is used to induce the model while the test set is used to evaluate the model (classifier). Additionally, the training set is further divided into two sets where one of them is called the *hold-out set*. The hold-out set is used to fine-tune the parameters for the learning algorithm. Since different splits of the example-objects could result in different models with different performance, the procedure described above should be repeated for a number of times. A systematic approach to this is the so-called *k-fold cross validation* in which the set is divided into k different disjunct subsets and where each of these subsets is used as a test set once. One should always remember that whatever methodology is used it will fail if the set of available example-objects does not satisfy the basic assumption of all inductive machine learning; the available data set should reflect the universe of objects that the system is to be operating on in the future.

4.2.3 The Relationship between Unsupervised Learning and Supervised Learning

In this chapter we have considered unsupervised learning as being a transformation of an information system into a decision system. As a consequence we can do supervised learning on the data set resulting from the unsupervised learning. In this way we do more than just finding natural groupings in the data set. We also induce a model that highlights the differences between the obtained groups and that can be used to classify unseen future objects. This may be useful in many applications and in particular in computational biology as described in Chapter 1 and illustrated in Figure 1.1.

EXAMPLE 4.2.3 (UNSUPERVISED LEARNING + SUPERVISED LEARNING)
Remember the clusters obtained from the information system in Table 4.1 by applying the unsupervised learning method in Example 4.2.1:

$$\{x_1\}_A, \{x_2\}_B, \{x_3, x_5\}_C, \{x_4, x_7\}_D, \{x_6\}_E, \{x_8\}_F$$

These clusters now determine a decision system $\mathcal{A} = (\{x_1, x_2, ..., x_8\}, \{$*Wine district, Main grape variety, Vintage, Storage temp.*$\} \cup \text{Cluster})$ *where* Cluster *is the decision attribute such that* Cluster *:* $U \rightarrow \{A, B, C, D, E, F\}$*. A wine expert can associate a Drink now / Hold-label to each of these clusters:*

$$\{x_1\}_{\textit{Drink now}}, \{x_2\}_{\textit{Hold}}, \{x_3, x_5\}_{\textit{Drink now or Hold}}, \{x_4, x_7\}_{\textit{Drink now}}, \{x_6\}_{\textit{Hold}}, \{x_8\}_{\textit{Hold}}$$

These new corrected labels determine the same decision system as shown in Table 4.2. Hence we can induce a model as described in Example 4.2.2. This model is shown in Table 4.3 and holds both descriptive and predictive knowledge as discussed earlier in this chapter.

In this small example the unsupervised learning seems rather unnecessary. The wine expert could easily inspect the eight wines without the clusters. However, for large data sets the advantage of working with groups of similar objects rather than single objects is vital in order for an expert to inspect and label the data set. □

Chapter 5

Similarity Measures and Syntactical Clustering

Given an information system $\mathcal{A} = (U, A)$, the objective of unsupervised learning is to search for ”natural” groups of similar objects called *clusters*. Consequently, we want to find a partitioning of the universe $U = \{X^1_{\mathcal{A}}, X^2_{\mathcal{A}}, ..., X^m_{\mathcal{A}}\}$. This requires both a *similarity measure*, which defines similarity between two objects, and a *clustering algorithm*, which specifies how to use the similarity measure in order to obtain good clusters.

Clustering algorithms can take two conceptually different approaches. We will call this approaches *blind* or *syntactical clustering* and *knowledge-based* or *semantical clustering*. The first approach searches blindly for clusters without any predefined knowledge. The second approach requires a specification of what to search for in advance. In addition, we can classify clustering algorithms according to weather they define crisp clusters or weather they define approximate clusters. Also they can be classified according to whether they define disjunct clusters or whether they define overlapping clusters. In this chapter we will be investigating blind algorithms that define crisp disjunct clusters. In the next chapter we will look into knowledge-based clustering.

Clustering is very much related to pattern recognition and is for this reason often treated in pattern recognition literature e.g. [Johnson and Wichern, 1998], [Schalkoff, 1992] and [Ripley, 1996].

5.1 Similarity Measures

Measures of similarity are fundamental to all clustering tasks. Consider an information system $\mathcal{A} = (U, A)$. The *information vector* is denoted $Inf_B(x) =< a(x) : a \in B >$, where $B \subseteq A$ and $x \in U$. Remember that that $a : U \to V_a$. We want to develop similarity measures that tell us something about the similarity between two objects $x \in U$ and $y \in U$ represented by their information vectors $Inf_B(x)$ and $Inf_B(y)$. In general, we want a similarity measure d that has the following property:

$$d(Inf_B(x), Inf_B(y)) = \begin{cases} 'large' & \text{when } x \text{ and } y \text{ belong to different clusters} \\ 'small' & \text{when } x \text{ and } y \text{ belong to the same cluster} \end{cases} \tag{5.1}$$

5.1.1 Similarity Measures in Vector Space

When dealing with numerical values the $n \times 1$ vector $Inf_B(x)$ can be interpreted as a point in the n-dimensional space. Hence distance can be interpreted as a measure of similarity. The *Euclidean (straight-line) distance* between two objects (represented by their information vector) is of the form

$$d(Inf_B(x), Inf_B(y)) = \sqrt{\sum_{a \in B} (a(x) - a(y))^2} \tag{5.2}$$

The related *"city-block" distance* is given by

$$d(Inf_B(x), Inf_B(y)) = \sum_{a \in B} (a(x) - a(y))^2 \tag{5.3}$$

Actually, both the Euclidean distance and the "city-block" distance are a special case of the more general *Minkowski metric*

$$d_p(Inf_B(x), Inf_B(y)) = \left[\sum_{a \in B} (a(x) - a(y))^2\right]^{1/p} \tag{5.4}$$

which for p = 1 becomes the "city-block" distance and for p = 2 becomes the Euclidean distance. Commonly, these distances are weighted (*weighted distances*). One example is

$$d_R^2(Inf_B(x), Inf_B(y)) = (Inf_B(x) - Inf_B(y))^T R(Inf_B(x) - Inf_B(y)) \tag{5.5}$$

in which R contains weights for the different attributes in B.

According to [Johnson and Wichern, 1998], two other popular distance measures are

$$\text{Canberra metric: } d_{cm}(Inf_B(x), Inf_B(y)) = \sum_{a \in B} \frac{|a(x) - a(y)|}{(a(x) + a(y))} \tag{5.6}$$

$$\text{Czekanowski coefficient: } d_{cc}(Inf_B(x), Inf_B(y)) = 1 - \frac{2\sum_{a \in B} min(a(x), a(y))}{\sum_{a \in B}(a(x) + b(y))} \tag{5.7}$$

5.1.2 Similarity Measures for Sets and Strings

When dealing with non-numerical values we often need measures for set and string similarity. Consider two sets $Inf_B(x)$ and $Inf_B(y)$ (Note that these sets, we might call them *information sets*, is simply the set of all elements in the information vector: $Inf_B(x) = \{a(x) : a \in B\}$ where $B \subseteq A$ and $x \in U$). Some examples of similarity measures discussed in [Schalkoff, 1992] are

$$d(Inf_B(x), Inf_B(y)) = \frac{|Inf_B(x) \cap Inf_B(x)|}{|Inf_B(x) \cup Inf_B(y)|} \tag{5.8}$$

$$d(Inf_B(x), Inf_B(y)) = \frac{|Inf_B(x) \cap Inf_B(y)|}{|Inf_B(x)| + |Inf_B(y)| - |Inf_B(x) \cup Inf_B(y)|} \tag{5.9}$$

$$d_L(Inf_B(x), Inf_B(y)) = max\{|Inf_B(x)|, |Inf_B(y)|\} - |Inf_B(x) \cap Inf_B(y)| \tag{5.10}$$

$d_L(Inf_B(x), Inf_B(y))$ is called the *Levenshtien distance.*

Unlike sets, the ordering of elements is important when comparing strings. Given two strings u and v, similarity could be based on the following:

- Inclusion: does string u contain v (or vice versa)?
- Overlap: finding the size of the largest substring in both u and v.
- Variational similarity: determining similarity on the bases of the minimum cost of ”converting” one string into another.

5.1.3 Similarity Measures for Time Series

A $n \times 1$ information vector $Inf_B(x)$ can also be interpreted as a time series, that is, each element is a measurement at time t_i such that $t_i > t_{i-1}$. The vector can now be represented as a two-dimensional graph. When comparing time series we often want to use similarity measures that compare gradients. The easiest way to do this is to transform the data into a language that posses this feature and then use the distance measures discussed above. One such transformation could simply be the computation of gradients:

$$\begin{aligned} Inf_B(x) = &< a_{t_1}(x), a_{t_2}(x), ..., a_{t_n}(x) > \xrightarrow{T} \\ Inf_B^T(x) = &< \frac{a_{t_2}(x) - a_{t_1}(x)}{t_2 - t_1}, \frac{a_{t_3}(x) - a_{t_2}(x)}{t_3 - t_2}, ..., \frac{a_{t_n}(x) - a_{t_{n-1}}(x)}{t_n - t_{n-1}} > \end{aligned} \tag{5.11}$$

More sophisticated transformations includes various mathematical transformation like the *Fourier Transformation* and the *Haar Wavelet Transformation* ([Struzik and Siebes, 1999]). In 1822 the French mathematician J. Fourier showed that any periodic function can be expressed as an infinite sum of periodic complex exponential functions. Later this principle was extended also to include discrete periodic functions or time series. The Fourier Transformation is a transformation from the time-domain to the frequency-domain. This is often useful in time series analysis since we are interested in the changes of the time series. However, while no frequency information is available in the time-domain, no time information is available in the Fourier transformed frequency-domain. This is a problem because we want to know *when* the changes occurred, not only *which* changes that occurred. The wavelet transform is a solution to this problem. It is capable of providing the time and frequency information simultaneously, hence giving a time-frequency representation of the signal. The Haar Wavelet Transformation is explained in more detail below.

The Haar Wavelet Transformation is an inner product of the time series, f, with a scaled and translated wavelet $\psi(x)$. This wavelet is most often the n-th derivative of some smoothing kernel $\theta(x)$. The choice of the smoothing kernel depends on the application and the desired properties of the wavelet transform. The scaling and translation action are performed by the scale parameter s, which changes the frequency content, and the parameter b, which determined the location of the analysing wavelet:

$$Wf(s,b) =< f, \psi > (s,b) = \frac{1}{s} \int_{\Omega} f(x)\ \psi(\frac{x-b}{s})\ dx \tag{5.12}$$

Here Ω is the length of the time series. The Haar Wavelet Transformation decomposes the time series into Haar components. These components describe the change (frequency) in the series at different times, and together they add up to the original series. Figure 5.1 shows the decomposition of an example time series. Decomposition is very useful when

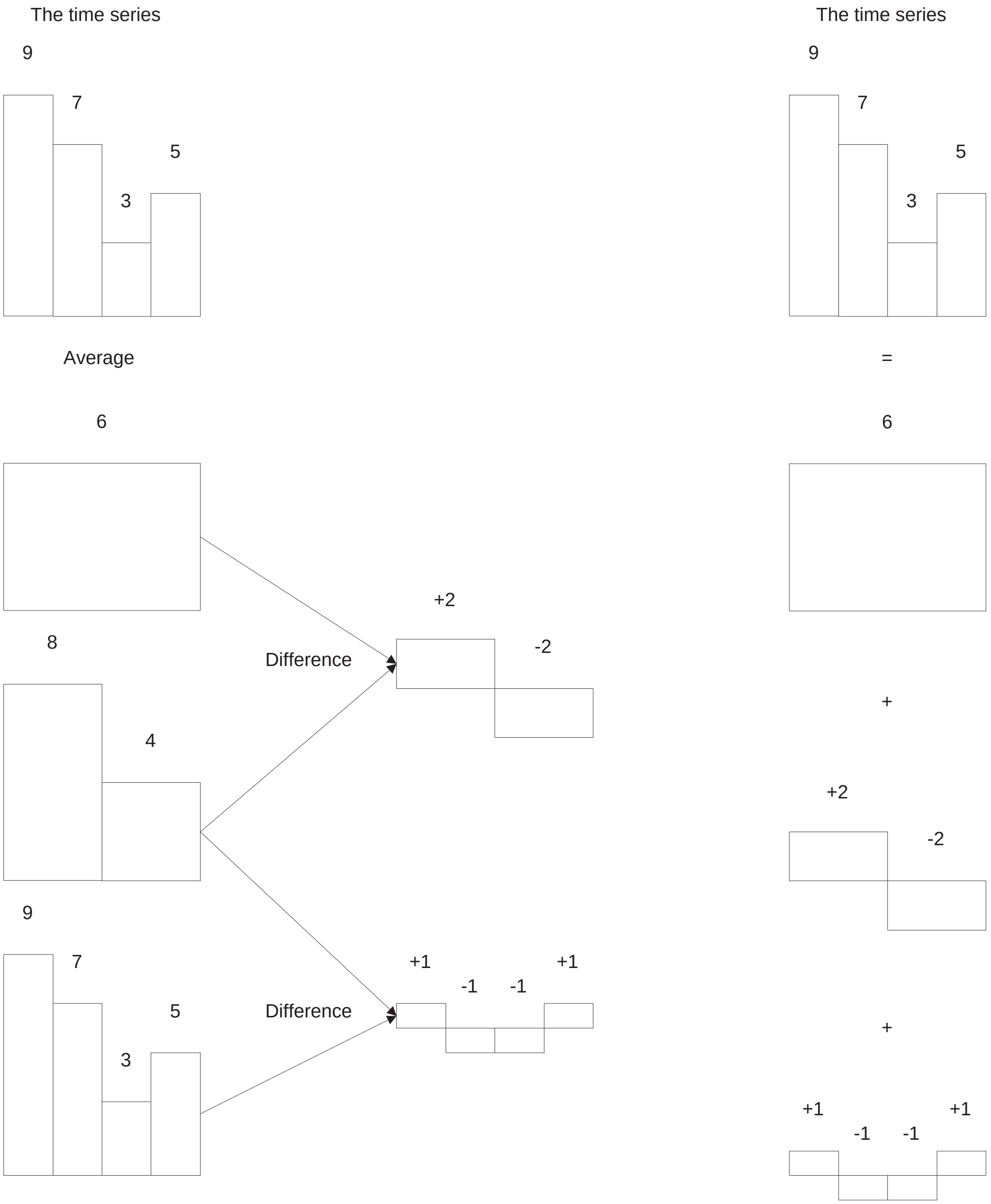


Figure 5.1: A practical application of the Haar Wavelet Transformation: Decomposition of an example time series into Haar components.

analysing time series since different components can be weighted differently in the similarity measure. For example, one most likely want to skip the constant component of the transformation when interested in comparing local slopes of two time series. In the same way one would certainly want to pay less attention to the high frequency components than to the low frequency components when designing similarity measures that compare the large trend in two time series. These are just examples of how the Haar Wavelet Transformation can be used as a tool in order to customise the similarity measure to a particular task.

Discretisation is also a very useful tool in the analysis of time series. It enables us to skip details by mapping a set of values with the same interpretation into the same unique value. One simple example is the *sign representation* of time series:

$$a_{t_i}(x) = \begin{cases} 1 & \text{if } a_{t_{i+1}} - a_{t_i} > 0 \\ -1 & \text{otherwise} \end{cases} \tag{5.13}$$

Note that discretisation often results in objects with non-numerical attributes. These categorical attributes are often represented as strings and thus we might need the similarity measures for sets and strings discussed above. For a thorough discussion on discretisation see [Nguyen and Skowron, 1995] and [Nguyen and Nguyen, 1998].

Transformations and discretisation are what we call *preprocessing* tools. In principle they could be an integrated part of the similarity measures. However, these calculations can be done only once since they are independent on which two objects to be compared. Thus in terms of performance it is better to do these calculations before any clustering algorithm is applied to the data. Note that by preprocessing we here mean similarity-specific calculations. Other preprocessing tasks, such as data cleaning etc. discussed in relation with the KDD process in Chapter 3, are excluded. Note that the ROSETTA system supports a wide variety of preprocessing tools including different discretisation algorithms.

5.1.4 Boolean Similarity Measures

So far we have discussed similarity measures of the form $d : U^2 \rightarrow \Re$. However, similarity measures can also take the form $d_B : U^2 \rightarrow \{0,1\}$. The indiscernibility relation is one example. This relation views similarity as a yes/no question rather than a task of assigning a value that represents a degree of similarity. This can be useful, especially when working with domain experts. They often find it easier to specify what significantly distinguishes two objects rather than to specify a mathematical measure of ”distance”. Hence, in many cases it can be easier to include domain knowledge in Boolean similarity measures than in traditional measures. Of course, a Boolean similarity measure could alternatively be defined by a “distance” measure d and a threshold τ:

$$d_{\mathrm{B}}(Inf_B(x), Inf_B(y)) = \begin{cases} \text{'indiscernible'} & \text{if } d(Inf_B(x), Inf_B(y)) < \tau \\ \text{'discernible'} & \text{otherwise} \end{cases} \tag{5.14}$$

5.1.5 The Similarity Matrix

Given an information system $\mathcal{A} = (U, A)$ and all pairs of objects $x_i, x_j \in U$, a similarity measure d determines a $|U| \times |U|$ *similarity matrix* with entry s_{ij}:

$$s_{ij} = d(Inf_B(x_i), Inf_B(x_j)) \tag{5.15}$$

The similarity matrix contains the information of similarity between all pairs of objects and is used both in the hierarchical clustering algorithms and in the indiscernibility-based clustering algorithm discussed in the next section.

5.2 Clustering Algorithms

We will define a clustering algorithm to be a strategy for applying a similarity measure to a data set in order to reveal the underlying natural groups. When developing algorithms one could always consider the search space. Given an information system $\mathcal{A} = (U, A)$ there are

$$\frac{m^{|U|}}{m!} \tag{5.16}$$

possible partitions of the universe into m nonempty subsets (clusters). Hence a data set containing 500 objects represent $10^{500}/10! = 2.756 \times 10^{493}$ possible ways of partitioning U into 10 clusters. Clearly, we can rule out any exhaustive search procedure. For this reason a wide variety of clustering algorithms have emerged that find ”reasonable” clusters without having to look at all configurations.

In this section we will examine algorithms that define crisp disjunctive clusters. We will divide these algorithms into *hierarchical clustering algorithms* and *non-hierarchical clustering algorithms*. The latter also includes *neural* and *Boolean approaches*.

5.2.1 Hierarchical Clustering Methods

Hierarchical clustering techniques proceed by either a series of successive merges or a series of successive divisions. *Agglomerative hierarchical methods* start with the individual objects, while *divisive hierarchical methods* work in the opposite direction. More precisely, the former starts with each object being a cluster of its own. Successively the two most similar clusters a merged until all clusters are fused into a single cluster. The latter starts with all objects in one cluster and successively divides the clusters into two sub-clusters such that the objects in one sub-cluster is ”far from” the objects in the other. This process is continued until each object forms its own cluster. The resulting clusters from both methods constitute a binary tree where the leaf nodes are single objects, the top node is a cluster containing all objects and the intermediate nodes are clusters containing the union of the clusters in the two children nodes. A diagram representing such a binary tree graphically is called a *dendrogram*.

Agglomerative hierarchical methods: Given an information system $\mathcal{A} = (U, A)$, the agglomerative hierarchical algorithm proceeds in the following manner:

1. Start with $n = |U|$ clusters, each containing one object, and a $|U| \times |U|$ similarity matrix $S = \{d_{ij}\}$ determined by a similarity measure d.
2. Search the similarity matrix for the most similar pair of clusters V and W. Let the distance between V and W be denoted d_{VW}.
3. Merge clusters V and W, and label the newly formed cluster (VW). Update the similarity matrix by (a) deleting the rows and columns corresponding to clusters V and W and (b) adding a row and a column that define the similarity between (VW) and the remaining clusters.
4. Repeat Steps 2 and 3 a total of $n - 1$ times.

Note that the similarity measures from the previous section only define similarity between two objects while the algorithm above demands methods for calculating similarity between clusters. Fortunately , methods for calculating similarity between clusters can easily be obtained from the ”single-object” similarity measures:

Single linkage: the similarity between two clusters is defined as the similarity between the most similar pair of objects (the two nearest neighbours) constituted of one object from each cluster.

Complete linkage: the similarity between two clusters is defined as the similarity between the most dissimilar pair of objects constituted of one object from each cluster.

Average linkage: the similarity between two clusters is defined as the average similarity between all pairs of objects constituted of one object from each cluster.

Divisive hierarchical methods: Given an information system $\mathcal{A} = (U, A)$, the divisive hierarchical algorithm proceeds in the following manner:

1. Start with a cluster containing all objects in the universe and let this cluster represent the top node of the future binary tree.
2. Choose a cluster Q represented by a temporary leaf node (thus this cluster contains more than one object) in the binary tree. Compute a $m \times m$ similarity matrix $S = \{d_{ij}\}$ determined by a similarity measure d (using one of the linkage methods defined above). Here each entry is the similarity between two clusters resulting from splitting Q in two and m is thus the possible number of ways of splitting Q into two non-empty clusters.
3. Search the similarity matrix for the most dissimilar pair of clusters V and W. Let the distance between V and W be denoted d_{VW}.
4. Divide Q into V and W such that $Q = V \cup W$.
5. Repeat Steps 2, 3 and 4 a total of $n - 1$ times.

Hierarchical clustering provides no methods for reallocation of objects that may have been incorrectly clustered at an early stage. Consequently, the method is sensitive to noise and in general the final set of cluster may not be optimal. The agglomerative hierarchical algorithm is clearly a $O(n^2)$ method. However, the divisive hierarchical method has no way of ”reusing” the similarity matrix from iteration to iteration. The similarity matrix has to be recalculated each time. Moreover, the size of the similarity matrix required to split a cluster with n objects into two non-empty clusters becomes $(2^{n-1} - 1) \times (2^{n-1} - 1)$. Thus the time complexity of the divisive hierarchical clustering algorithm described above is exponential. For this reason, other methods are normally used to split each cluster in two, rather than looking at all configurations. As a conclusion we might say that the hierarchical clustering strategy is a rather time consuming method and is better used on relatively small data sets.

Finally, we will look at the *Ward's hierarchical clustering method* which can either be implemented as a agglomerative or a divisive hierarchical method. Ward considered hierarchical clustering procedures based on minimising or maximising the ”loss of information” from joining or splitting clusters. The ”loss of information” is often interpreted as the *sum of squared error (SSE)*. Let $U = \{X^1_{\mathcal{A}}, X^2_{\mathcal{A}}, ..., X^m_{\mathcal{A}}\}$ be a partition of the information system $\mathcal{A} = (U, A)$ and let n_i be the number of objects in cluster $X^i_{\mathcal{A}}$:

$$Inf_B(c_i) = \frac{1}{n_i} \sum_{x \in X^i_{\mathcal{A}}} Inf_B(x) \tag{5.17}$$

$$SSE = \sum_{i=1}^{m} \sum_{x \in X^i_{\mathcal{A}}} \sqrt{\sum_{a \in B} (a(x) - a(c_i))^2} \tag{5.18}$$

$Inf_B(c_i)$ is called the *centroid* of cluster $X^i_{\mathcal{A}}$. SSE gives us the sum of the total variance within each cluster of the given partition. It can be shown that for an agglomerative implementation of the Ward's hierarchical clustering the smallest increase in SSE results from merging the pair of clusters represented by their centroids $Inf_B(c_i)$ and $Inf_B(c_j)$ for which the measure

$$L_{ij} = \frac{n_i n_j}{n_i + n_j} \sum_{a \in B} (a(c_i) - a(c_j))^2 \tag{5.19}$$

is minimised. Ward's hierarchical clustering method proceeds in exactly the same manner as the hierarchical methods defined earlier except for the fact that the SSE value is used to determined the ”similarity matrix” rather than a “normal” similarity measure.

The definition of SSE gives us a tool for defining a stop criterion for the hierarchical algorithm which determines a fixed number of clusters m:

$$f(SSE_m, SSE_{m+1}) < \tau \tag{5.20}$$

where f is some function; $f(x, y) = x - y$, $f(x, y) = \frac{x}{y}$, etc. Of course, a value for τ still needs to be found.

5.2.2 Non-hierarchical Clustering Methods

Non-hierarchical clustering methods start from either an initial partition of objects into clusters or an initial set of seed points (centroids for the resulting clusters). Normally the number of clusters is specified in advance. Since the similarity matrix does not have to be determined, non-hierarchical methods can be applied to much larger data sets than can hierarchical techniques.

The K-means algorithm: This is one of the most popular algorithms in which each object is assigned to the cluster with the nearest centroid. The algorithm is composed of the following simple steps:

1. Partition the information system into k initial clusters.
2. Scan through the universe of objects and assign each object to the cluster whose centroid is nearest (using some similarity measure). Recalculate the centroid for all clusters that received or loosed any objects.
3. Repeat Step 2 until no more reassignments take place.

Equivalently one could start with specifying k initial centroids and then proceed to Step 2.

A neural approach: Self-Organising Maps: Self-Organising Maps ([Kohonen, 1990]) is a sheet-like artificial neural network where the cells become specifically tuned to various objects through an unsupervised learning process. Self-Organising Maps bear strong resemblance to both the k-means algorithm and to *Learning Vector Quantisation(LVQ)*. The algorithm consists of the following steps, were each iteration is indexed by k:

1. Select an initial set of centroids and denote them $Inf_B(c_i)$.
2. For each object $x \in U$, find the closest centroid (using some similarity measure) and denote it $Inf_B(c_c)$.
3. Update $Inf_B(c_c)(k)$ to form $Inf_B(c_c)(k+1)$ as follows:

$$Inf_B(c_c)(k+1) = Inf_B(c_c)(k) + \alpha(k)[x - Inf_B(c_c)(k) \quad (5.21)$$

$$Inf_B(c_i)(k+1) = Inf_B(c_i)(k) \text{ for } i \neq c \quad (5.22)$$

4. Repeat Step 2 and 3.

Here $\alpha(k)$ is an iteration-dependent parameter, $0 < \alpha(k) < 1$, that decreases monotonically with k. Hence $\alpha(k)$ helps controlling the convergence of the algorithm.

Both the k-means algorithm and the Self-Organising Maps algorithm are iterative algorithms that converge against some stable configuration. However, the final assignment of objects to clusters will in some degree depend on the initial selection of seed points. One should always rerun the algorithm with different initial configurations and compare the results. If the results differ significantly, it might indicate that the data set does not contain any stable configuration of natural groups. Alternatively, it might indicate that the selected number of initial seed points is unfavourable. Selecting too many initial seed points might result in the situation where two or more seed points lie within a single cluster and thus split an otherwise natural group. On the other hand, too few initial seed points will produce at least one cluster with very disperse objects. Even if the universe is known to consist of k natural groups, all these groups may not be represented in the available information system. For this reason, it is always a good idea to also rerun the algorithm for different numbers of initial seed points. Since the two non-hierarchical algorithms defined above both are $O(kn)$ methods (k being the number of iterations), they will still have a favourable time consumption compared to the hierarchical methods.

A Boolean approach: Indiscernibility-based clustering: Given a Boolean similarity measure (the indiscernibility relation), the indiscernibility-based clustering algorithm takes a very simple approach to clustering in that it specifies an indiscernibility graph where each node represents one object and two nodes are connected if the two respective objects are indiscernible. Now each separate graph in the full indiscernibility graph is considered a cluster.

The ROSETTA system implements the indiscernibility-based clustering approach and outputs a graph-specification that can be used together with the graph drawing software *Graphviz* (see [Graphviz, 1999]) to draw the indiscernibility graph.

The indiscernibility-based clustering approach somewhat escapes the problem of selecting a number of clusters in that the number of clusters follows as a direct consequence of the defined Boolean similarity measure. If this measure is defined restrictively, only a few objects will be connected in a few tight clusters accompanied by a large number of unclustered objects. If the measure is loosely defined, a large number of objects, possibly all, will be connected in a few big width-spread cluster. Consequently, the challenge is to define a Boolean similarity measure such that the algorithm results in a favourable number of distinguishable clusters that together contain a sufficiently large number of objects. Note that this algorithm not only defines a set of clusters, but also says something about the relationship between objects within the cluster. To conclude the discussion about indiscernibility-based clustering we should mention that the time complexity of this algorithm is $O(n^2)$ since it looks at all possible pair of objects in the universe.

5.3 Discussion: Finding the Best Clusters

Both hierarchical and non-hierarchical clustering algorithms considered in the last section have the property that not all partitions of the data set are considered. Consequently, they are attractive in that Equation (5.16) is considerably reduced. However, all the algorithms discussed also have the property that they can lead to a sub-optimal data set partitioning. The situation leading to a sub-optimal partitioning is either due to (a) an unfavourable number of clusters or, of course, (b) a distribution of the objects among the clusters that is not optimal. Optimising both the number of clusters and the distribution of objects among the clusters are not trivial since these two criteria are partly conflicting. Note, for example, that the SSE value (see Equation 5.18) equals zero (minimum) when each object forms its own cluster. However, this would hardly be considered a good partitioning. What we desire, of course, is a trade off between the number of clusters and the tightness of each cluster.

In general, we desire a partition of the objects contained in the information system $\mathcal{A} = (U, A)$ such that a specified *clustering function* J is minimised. Given two competing partitions P_1 and P_2 we want

$$J(P_1) < J(P_2) \tag{5.23}$$

if P_1 is considered to be a better partition than P_2. Note that SSE has this property. We can develop an iterative algorithm that incrementally reorganises an initial configuration of clusters by moving one object x from cluster $X^i_{\mathcal{A}}$ to cluster $X^j_{\mathcal{A}}$ such that J is reduced. This strategy can be used to fine-tune the clusters resulting from one of the algorithms (hierarchical / non-hierarchical) discussed above.

Finally, it is worth mentioning that one always should remember that the resulting clusters very much depend on the chosen similarity measure. Thus ”optimal” in this context is relative to the chosen definition of similarity.

5.4 Case Study: The Party Families in Europe 1970-1992

To illustrate the methods discussed in this chapter we will now use them on an example data set. The data set, depicted in Table 5.1, reflects the mean electoral support of the major party families in 17 different European countries from 1970 to 1992. It is taken from a book in comparative politics ([Heidar and Berntsen, 1995]) where its properties were discussed using non-computational methods. We will use three different clustering methods to find groupings of countries that have had a similar distribution of votes among the party families in the given period. The three methods considered are (1) the Ward's hierarchical clustering algorithm implemented as an agglomerative method, representing the hierarchical clustering methods, (2) the k-means algorithm, representing the converging non-hierarchical clustering methods and (3) the indiscernibility-based clustering algorithm.

1. The final assignment of countries to clusters resulting from using the Ward's hierarchical clustering algorithm can be seen in Figure 5.2. Using the stop criterion $SSE_m - SSE_{m+1} < 1000$ we obtain six cluster:

   a$_1$: Netherlands, Luxembourg, Belgium and Switzerland

   b$_1$: Germany, Austria and Italy

   c$_1$: Sweden, Denmark and Norway

   d$_1$: Greece and UK

   e$_1$: Spain, France and Portugal

   f$_1$: Iceland and Finland

   Clearly the clustering coincides somewhat with the countries geographical location and this could indicate that our clusters are fairly good.

2. Using the k-means algorithm with six initial clusters results in the following assignment of countries to clusters:

   a$_2$: Netherlands, Luxembourg, Belgium, Germany, Austria and Italy

   b$_2$: Switzerland

   c$_2$: Sweden, Denmark and Norway

   d$_2$: Greece, UK and Spain

   e$_2$: France and Portugal

   f$_2$: Iceland and Finland

   Comparing these clusters to the one obtained from using the hierarchical algorithm we see that clusters **a** and **b** are merged except for Switzerland which now forms a cluster of its own. Also Spain has moved from cluster **e** to cluster **d**.

Country	Communists	Socialists	Greens	Social Democrats	Liberals	Agrarians	Subnational, regional and ethnic parties	Christian Democrats	Conservatives	Extreme Right
Norway	0	7	0	38	4	8	0	9	24	6
Sweden	6	0	2	43	10	17	0	2	18	1
Denmark	4	9	0	33	13	14	0	3	15	9
Finland	15	0	2	24	3	25	5	3	21	0
Iceland	0	18	3	16	4	22	0	0	36	0
UK	0	0	9	39	15	0	4	0	42	0
Netherlands	2	5	0	30	23	0	0	37	0	0
Belgium	2	0	4	27	19	0	14	31	0	2
Luxembourg	6	1	3	31	21	0	0	34	0	1
Switzerland	2	2	7	22	23	11	0	22	3	5
Austria	1	0	2	48	0	0	0	41	0	8
Germany	1	0	3	40	9	0	0	46	0	1
France	15	2	2	28	20	0	0	0	25	5
Italy	29	0	3	15	4	0	3	35	2	6
Greece	10	0	0	39	6	0	0	0	44	0
Spain	8	0	0	39	16	0	10	0	21	0
Portugal	15	0	1	31	38	0	0	1	11	0

Table 5.1: Mean electoral support of the European party families from 1970 to 1992 by country (% of vote)

3. Using the indiscernibility-based clustering approach we define the similarity measure to consider two objects to be indiscernible if the Euclidean distance between them does not exceed 30. The result is graphically displayed as a indiscernibility graph in Figure 5.3. This graph defines five clusters (alternatively, we could say that it defines three clusters and leaves two objects unclustered):

   $\mathbf{a}_3$: Netherlands, Luxembourg, Belgium, Germany, Austria and Switzerland.

   $\mathbf{b}_3$: Sweden, Denmark, Norway, Iceland, Finland and UK

   $\mathbf{c}_3$: Portugal, Spain and Greece

   $\mathbf{d}_3$: Italy

   $\mathbf{e}_3$: France

   Again we see a strong resemblance to the clusters defined by the other algorithms. We can clearly see how the different algorithms merge and divide the same basic clusters to constitute clusters specific to this algorithm.

The k-means algorithm was implemented using the Euclidean distance as a similarity measure, while the hierarchical algorithm used SSE. However, the SSE (recall that SSE is an abbreviation for the Sum of Square Error) computed for a given cluster is really just the ”sum of the squared Euclidean distances between the cluster’s centroid and each object in that cluster”. Hence the two algorithms should be well suited for comparison. Also the indiscernibility-based clustering approach is in this example based on the Euclidean distance.

The k-means algorithms seemed to alternate between several different cluster configurations for different initial configuration. The clusters listed above are just one of them. This might indicate that the data set is not very well suited for finding good clusters, that is, natural groups where the internal distances are short and the distances to other clusters are long.

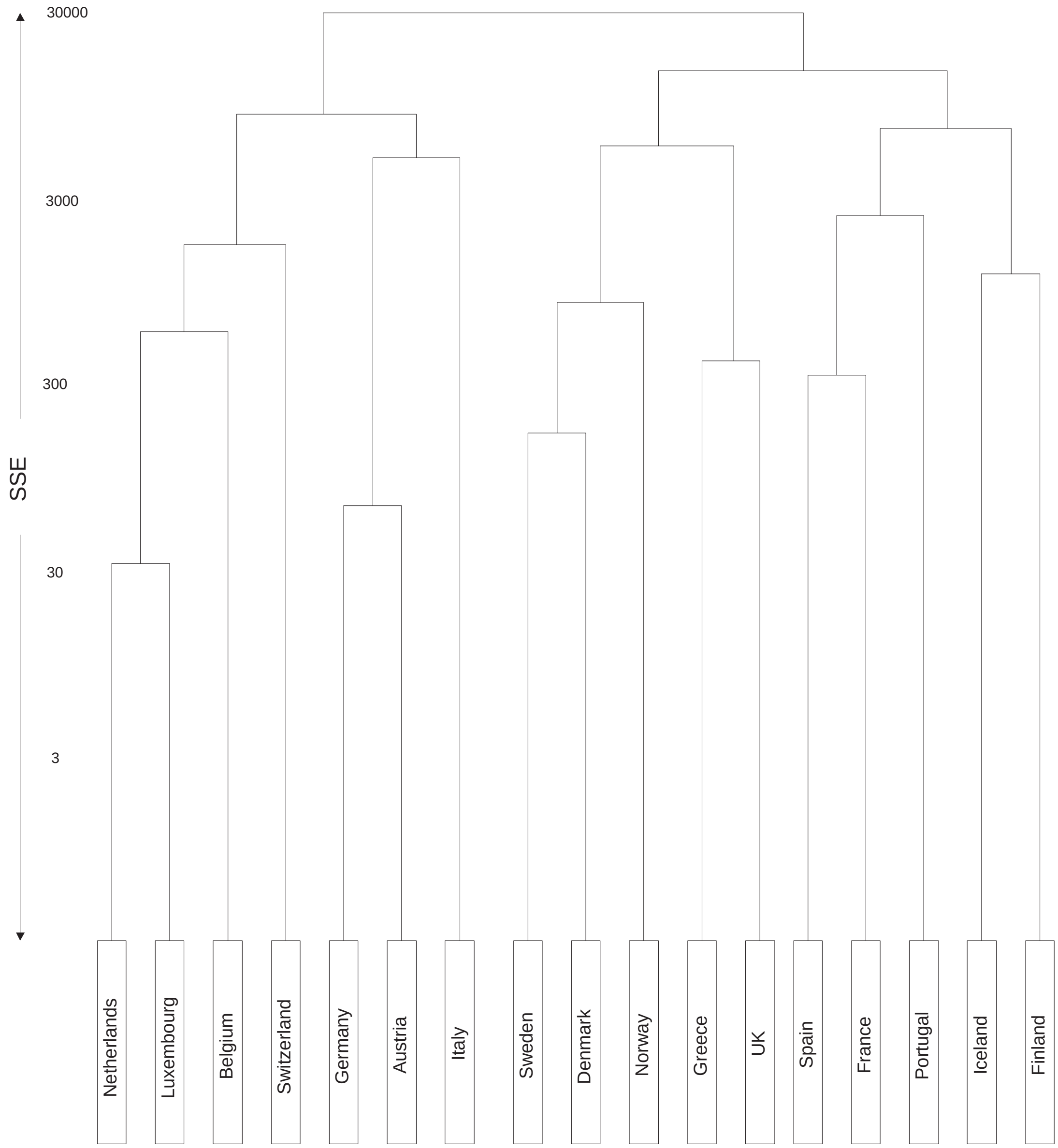


Figure 5.2: Dendrogram showing the resulting partition from applying the Ward's hierarchical clustering method to the data set in Table 5.1.

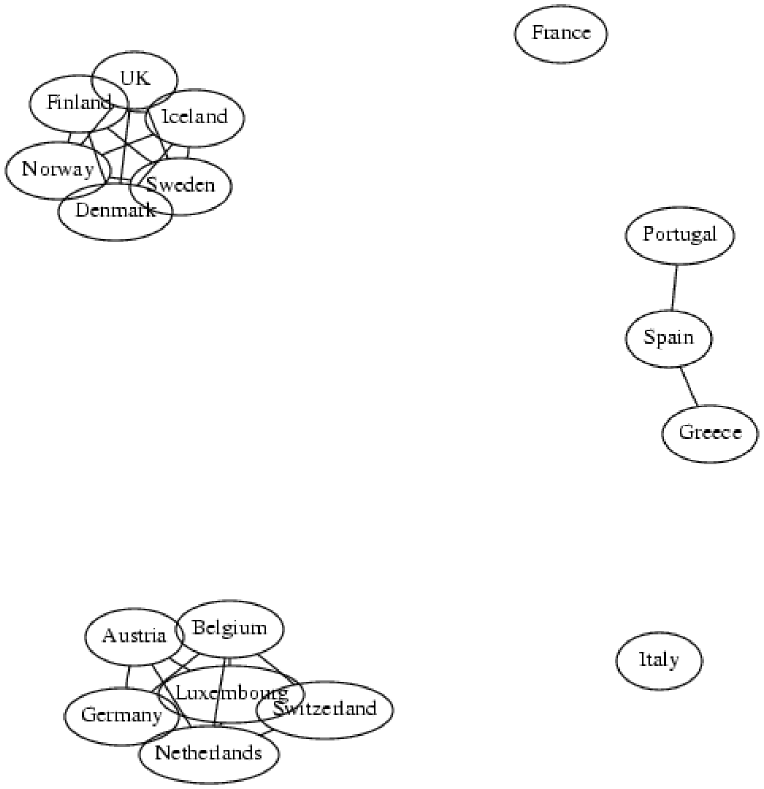


Figure 5.3: Indiscernibility graph showing the resulting partition from applying the indiscernibility-based clustering method to the data set in Table 5.1.

5.5 Time Series Case Study: Party competition

To illustrate the principle of applying clustering algorithms to time series we will now process the data set depicted in Table 5.2 The data is again taken from a book in comparative politics ([Gallagher et al., 1995]) where its properties were discussed using non-computational methods. This data set reflects the mean number of parties in competition in 15 European countries over the last four decades. We will apply the indiscernibility-based clustering method extended to include the following steps:

Countries	1950s	1960s	1970s	1980s
Austria	4	4.5	4	4.5
Belgium	5.7	6	9.3	12.3
Denmark	6.5	7.8	10.6	10.5
Finland	6.3	8	9.3	9.5
France	7	7.3	7.5	7.3
Germany(West)	7.5	4.7	3	4
Iceland	5.3	5	5	7.5
Ireland	5.3	4.3	4	5.3
Italy	8.5	8	8.7	10
Luxembourg	4.3	4.5	6.5	6
Netherlands	7.7	9.5	11.7	8.5
Norway	6	7	8	7
Sweden	5	6.3	6	7
Switzerland	9	9	10.3	10
United Kingdom	3	3	4.8	6

Table 5.2: The fragmentation of the European party system: Mean number of parties in competition from 1950 to 1990. Includes only those parties polling at least 1 percent.

1. Apply the Haar Wavelet Transformation to the data set:

$$\mathcal{A} = (U, A) \overset{HWT}{\rightarrow} \mathcal{A}_{HWT} = (U, A_{HWT}) \tag{5.24}$$

2. Discretise the transformed data using the sign representation below on each attribute $a \in A$ for each object $x \in U$:

$$a_{t_i}(x) = \begin{cases} 1 & \text{if } a_{t_{i+1}} - a_{t_i} > \tau \\ -1 & \text{if } a_{t_{i+1}} - a_{t_i} < -\tau \\ 0 & \text{otherwise} \end{cases} \tag{5.25}$$

   In this example $\tau = 0.6$

3. Define a suitable indiscernibility relation. In this example the constant Haar component was discarded and the relation was defined such that requirements for similarity was stricter for low frequency component than for high frequency components.

4. Use the indiscernibility based clustering algorithm on the transformed discrete data and display the result as a indiscernibility graph.

The Haar Wavelet Transformation represent a shift of focus from values to local slopes. Hence the obtained clusters are groups of countries with a similar evolution in terms of numbers of competing political parties. The results from applying the above method to the data set in Table 5.2 can be seen in Figure 5.4. The clusters presented as time series graphs can be seen in Figure 5.5.

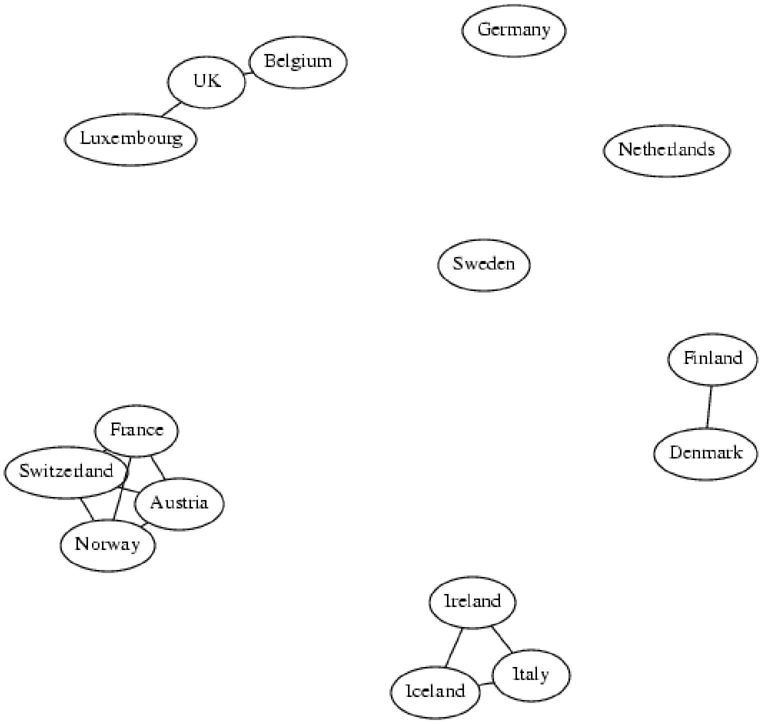


Figure 5.4: Indiscernibility graph showing the resulting partition from applying the indiscernibility-based clustering method to the data set in Table 5.2.

A political science discussion of the results from this and the previous section is outside the scope of this thesis.

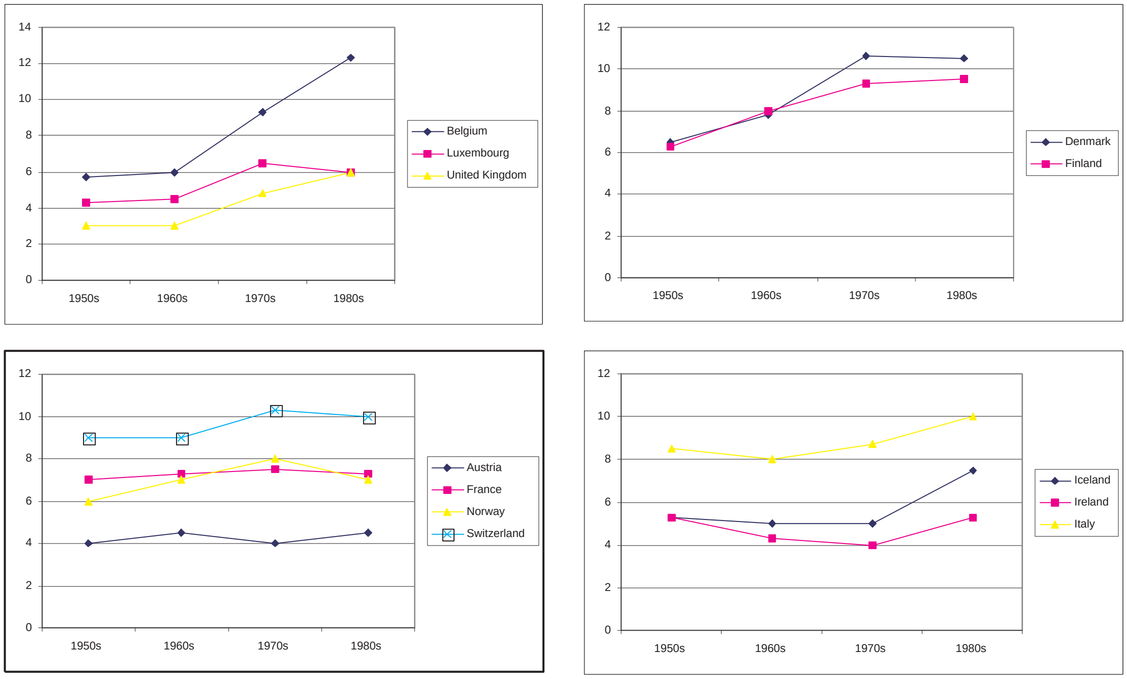


Figure 5.5: The clusters in Figure 5.4 were countries in the same clusters are displayed together as time series graphs in order to show their similarity.

Chapter 6

Knowledge-based Clustering

In Chapter 5 we explored different syntactical clustering approaches that blindly searched for similar objects in a data set. This approach seems intuitively attractive because of its generality and its ability to discover existing patterns without having to make any predefined assumptions. However, when co-operating with domain experts we often want to work by means of hypotheses. Hypotheses can of course be validated against the discovered clusters using methods from Chapter 5, but these clusters often prove difficult to interpret since they never quite seem to coincide with the expert's assumption or hypothesis. This problem can in many cases be dealt with by defining what we are looking for in advance. Using the methods in Chapter 5, domain knowledge can be included as a help in the search strategy either by defining the number of initial clusters for non-hierarchical methods, or even defining the locations of the seed points for these clusters, or by defining the similarity measure. However, even more explicit is the approach of predefining the features of each cluster and then simply distributing the objects among these clusters according to their match. We will denote this method *knowledge-based* or, more specifically, *template-based clustering*. A template is simply a piece of encoded biological knowledge that alone or as an element in a set of templates defines or even determines a cluster in that all objects matching this or these templates belong to the same cluster. One can argue that this approach is more related to pattern matching or classification than it is to clustering or unsupervised learning. However, we will use the term "clustering" in this thesis because in many aspects the knowledge-based clustering approach is an extension of syntactical clustering approach as argued above.

We again define the data set as an information system $\mathcal{A} = (U, A)$ and additionally consider the following definitions:

Object $x \in U$**:** The input pattern to the template matching process defined by its information vector $Inf_B(x)$.

Template t_i**:** The template determining cluster $X^i_{\mathcal{A}}$.

Range R: the extent of x over which the match occurs. This might simply be a subset of A denoted $C \subseteq A$ or a subset of the components resulting from some kind of transformation used on x.

Given the definitions above the following two candidate metrics indicating mismatch can be used:

$$m_1(t,x) = \sum_R |t - x| \tag{6.1}$$

$$m_2(t,x) = \sum_R (t - x)^2 \tag{6.2}$$

Intuitively, m_1 and m_2 will be small when t and x are similar, and large when they are significantly different. Obviously we would like to define a function that assigns x to cluster $X^i_{\mathcal{A}}$ if $m_j(t_i,x) < \tau$. Of course, the determination of τ is critical. τ needs to be selected with a view to statistical significance and also with a view to more subjective issues like at which degree of strictness we want our clusters to be defined.

Template-based clustering is discussed in the literature both in general and in connection with gene expression analysis. One example is [Carr et al., 1997]. Another example is [Lowe et al., 1999] which defines a fuzzy template T (to be used in the time series domain) as a triple

$$T = (\tau, S, \{T_1, ..., T_k\}) \tag{6.3}$$

where τ is a fuzzy interval defining the starting point of the template, S is a fuzzy segment and $\{T_1, ..., T_k\}$ is a list of k fuzzy sub-templates. Moreover, S is defined as the pair $S = (C, d)$ where C is a fuzzy course representing the allowable spread of the signal to be matched (objects are often called signals in the time series domain) and d is a fuzzy duration of time over which the segment is defined. The template matching can be performed via the calculation of a membership function $\mu \in [0, 1]$ where $\mu = 0$ means that the given signal do not match the fuzzy template to any degree and $\mu = 1$ means that the template is fully matched. Fuzzy approaches to template matching have some advantages in that they can easily handle incomplete data in terms of vagueness in the definition of intervals and signal level.

Template-based clustering is most often used in time series domains. This is because a template intuitively can describe a curve over a sub-time-interval and also that it makes sense to define such patterns in advance since we often are looking for signals with a certain response in a certain sub-interval.

6.1 Template-based Cluster Analysis in Time Series

In the following section we will look at an approach to template-based clustering in the time series domain. We will view the matching process as a Boolean function that decides whether or not an object matches a predefined template. We will define a template to be a basic curve feature accompanied by a set of requirements that needs to be fulfilled in order for an object to match this template. As an example, consider the template that describes the feature of monotonic increase over at least t time point and with a total increase of τ. Note that our definition does not describe a specific interval over which the template is defined. Instead we will develop an algorithm that looks at all possible sub-intervals of the time series. These intervals will together with the set of initially defined templates determine a set of clusters that describe all the features of all objects. We will then order the set of intervals according to the number of matches occurring in this interval. This algorithm consists of the following steps:

1. Start with the set of time series contained in the information system $\mathcal{A} = (U, A)$ and a set of templates T. The set of all possible sub-interval of the time series will be denoted I.

2. Build a new information system $\mathcal{A}_I = (U, I)$ where I is the set of attributes representing each possible sub-interval such that $i\ :\ U \to T \cup \emptyset$ for every $i \in I$. Consequently, each entry in the table is the template that matches the given object in the given time-sub-interval. If no template matches the object in that sub-interval, the entry is empty.

3. Simplify the new information system $\mathcal{A}_I = (U, I)$ such that:

$$i(x) = \begin{cases} \emptyset & \text{if it exists an } i_{super} \in I \text{ such that } i_{super}(x) \in T \text{ and } i \subset i_{super} \\ i(x) & \text{otherwise} \end{cases} \tag{6.4}$$

   Consequently, we have discarded any template that matches an object in a sub-interval of another template that includes the same feature.

4. Denote the ordered set of the intervals Int and initials it such that $Int = \emptyset$. As long as the information system $\mathcal{A}_I = (U, I)$ contains any object:

   - Find the interval $i_{best} \in I$ in which the most objects matches a template.
   - Remove i_{best} from the information system together with every object that no longer match any template in any sub-interval.
   - Add i_{best} to Int.

5. Now the set of intervals Int and the set of templates T together determine a set of overlapping clusters that describe the objects according to the pre-defined templates.
   Note that Step 4 alternatively can be executed separately for each template. Then we will end up with a collection of sets of intervals where each set of intervals is relative to a specified template. Using this strategy, the stop criterion in Step 4 becomes "for each template $t \in T$ continue as long as the information system contains any objects matching template t". In addition, the information system needs to be initiated for each new template.

The algorithm explained above gives us a description of the data set $\mathcal{A} = (U, A)$ according to the pre-defined templates. The set Int contains all the intervals in which one or more objects matches a template (after simplification). Since the set is ordered we can pick a subset of intervals from Int such that a sufficiently large part of the universe of objects is described. When a good set of clusters are found, we can add a decision attribute to the simplified information system $\mathcal{A}_I = (U, I)$ and use the rough set framework described in Chapter 4 to find minimal sets of intervals that are needed to describe these clusters.

6.2 Time Series Case Study Using Templates: Party competition

We will pick up the thread of the example time series depicted in Table 5.2 (Chapter 5) and now analyse it using the template based algorithm described in this chapter. We are interested in grouping the countries according to whether they have experienced a significant change in the number of competing parties or not. Thus we define two templates; (I) significant increase and (D) significant decrease. We will define "significant" as being an increase or a decrease in the number of competing parties of at least 1 over a period of at least two decades.

Table 6.1 shows the different possible sub-intervals of the time series and which templates matching the different countries in these intervals. Table 6.2 shows the same table after simplification. Note how the entries for Belgium, Finland and United Kingdom have changed.

Countries	50s - 70s	50s - 80s	60s - 80s
Austria			
Belgium	I	I	I
Denmark	I		
Finland	I	I	I
France			
Germany(West)	D		
Iceland			I
Ireland	D		
Italy			I
Luxembourg	I		
Netherlands	I		
Norway	I		
Sweden			
Switzerland	I		
United Kingdom	I	I	I

Table 6.1: Table shows the different possible sub-intervals and which template matches the countries in these intervals. 'I' stands for "increasing" and 'D' stands for "decreasing".

Countries	50s - 70s	50s - 80s	60s - 80s
Austria			
Belgium		I	
Denmark	I		
Finland		I	
France			
Germany(West)	D		
Iceland			I
Ireland	D		
Italy			I
Luxembourg	I		
Netherlands	I		
Norway	I		
Sweden			
Switzerland	I		
United Kingdom		I	

Table 6.2: Table showing a simplified version of Table 6.1 where all redundant information is removed. 'I' stands for "increasing" and 'D' stands for "decreasing".

From this analysis we can order the intervals as follows:

1. ’50s - 70s’ includes 7 matches
2. ’50s - 80s’ includes 3 matches
3. ’60s-80s’ includes 2 matches

Note that the countries that did not match any template in any interval actually had no significant change in the number of competing parties from the 50s to the 80s. Thus one relevant partitioning of countries into clusters would simply be to group the countries that experienced a significant change in one cluster and to group the countries that did not experienced a significant change in another cluster. Alternatively, one could split the cluster containing those countries that experienced a change into those countries who experienced an increasing number of parties and those countries who experienced a decreasing number of parties. An even more fine-tuned partition would be to take into consideration *when* the change occurred, that is, to further divide the clusters of countries that experienced a change into those who experienced the change during the same time sub-interval.

Part III

Computational Biology Case Studies

In this part we will apply some of the clustering algorithms described in Part II on real world gene expression data sets in order to extract useful knowledge as explained in Part I. In particular, we will use our indiscernibility-based clustering approach from Chapter 5 together with the template-based method from Chapter 6 in a detailed analysis of the Fibroblast data ([Iyer et al., 1999])

Chapter 7

The Fibroblast Data

In this chapter we will present a thorough cluster analysis of the Fibroblast data described in [Iyer et al., 1999]. Two different clustering approaches will be used for this purpose. The first approach is the syntactical clustering method which we called indiscernibility-based clustering in Chapter 5. The second approach is the template-based clustering method described in Chapter 6. The approaches applied to the Fibroblast data are published in [Hvidsten et al., 1999b] and [Hvidsten et al., 2000] respectively.

7.1 The Data Set

As briefly described in Chapter 3, [Iyer et al., 1999] studies the human fibroblast response to serum which appears to be related to the physiology of wound repair. The temporal changes in mRNA level of 8613 human genes were measured at 12 times ranging from 0 minutes to 24 hours after serum stimulation. A subset of 517 genes whose expression changed substantially in response to serum was selected for further analysis (this data set is publicly available on the WEB: $< \mathtt{http}:// \mathtt{genome} - \mathtt{www.standford.edu/serum} >$).

[Iyer et al., 1999] uses an agglomerative implementation of the hierarchical clustering method to cluster the 517 genes into groups on the basis of the similarity of their expression profiles over the full 24 hours. Ten clusters were identified containing 452 of the 517 genes.

Figure 7.1 shows a segment, the ten first genes, of the data set. Three different aspects are worth mentioning about the data itself. Firstly, the data is normalised at time zero. As a consequence the data set describes the reaction of genes to serum stimulation relative to their initial expression level. Secondly, the data is logarithmic. Hence, the genes have the initial value of 1 and always stay above zero. Thirdly, the sampling rate is gradually reduced with time. Therefore, the interval between two arbitrary measurement points is not necessarily constant.

It is important to be aware of the uncertainties in the data material. It would be rather naive to believe that the measurements resulting from a microarray experiment are 100% exact. In the Fibroblast data there are several genes that are measured more than one time. Figure 7.2 illustrates the difference in the measured expression level for these genes. Although the number of measurements in this case is two small to draw any valid statistical conclusions, the graphs in Figure 7.2 give us a hint about what kind of noise and uncertainty we are dealing with.

Gene	0HR	15MIN	30MIN	1HR	2HR	4HR	6HR	8HR	12HR	16HR	20HR	24HR
1	1.00	0.72	0.10	0.57	1.08	0.66	0.39	0.49	0.28	0.50	0.66	0.52
2	1.00	1.58	1.05	1.15	1.22	0.54	0.73	0.82	0.82	0.90	0.73	0.75
3	1.00	1.10	0.97	1.00	0.90	0.67	0.81	0.88	0.77	0.71	0.57	0.46
4	1.00	0.97	1.00	0.85	0.84	0.72	0.66	0.68	0.47	0.61	0.59	0.65
5	1.00	1.21	1.29	1.08	0.89	0.88	0.66	0.85	0.67	0.58	0.82	0.60
6	1.00	1.45	1.44	1.12	1.10	1.15	0.79	0.77	0.78	0.71	0.67	0.36
7	1.00	1.15	1.10	1.00	1.08	0.79	0.98	1.03	0.59	0.57	0.46	0.39
8	1.00	1.32	1.35	1.13	1.00	0.91	1.22	1.05	0.58	0.57	0.53	0.43
9	1.00	1.01	1.38	1.21	0.79	0.85	0.78	0.73	0.64	0.58	0.43	0.47
10	1.00	0.85	1.03	1.00	0.81	0.82	0.73	0.51	0.24	0.54	0.43	0.51
...	...	...	...	...	...	...	...	...	...	...	...	...

Figure 7.1: The ten first genes in the original Fibroblast data.

7.2 Approach 1: Indiscernibility-based Clustering Analysis

The first approach is more or less the same as the one taken by [Iyer et al., 1999] in that we cluster the genes into a few large clusters one the bases of their similarity over the full 24 hour period. The following method were used:

Linearisation: The data was linearised by applying the logarithmic transformation log_2 to each data point (for results see Figure 7.3). This is due to the fact that linearised values can be compared on an equal basis anywhere on the value scale.

Haar Wavelet Transformation: This transformation shifts the focus from expression levels to local slopes and makes it easier to customise a similarity measure with the wanted properties. Note that the Fibroblast data set has 12 attributes (which is not of the form 2^n). As a consequence we needed to do some approximations to the transformation. Two different approaches were used and can be seen in Figure 7.4 and Figure 7.5. The first approach is the most correct since the Haar components sum up to the original time series. The second approach has lost this property, but at the same time it possesses the favourable property of having fewer attributes. In the following analysis the transformation in Figure 7.5 will be used.

Discretisation: The transformed data was discretised using the sign representation below on each time point for each gene:

$$a_{t_i}(x) = \begin{cases} 1 & \text{if } a_{t_{i+1}} - a_{t_i} > 0.2 \\ -1 & \text{if } a_{t_{i+1}} - a_{t_i} < -0.2 \\ 0 & \text{otherwise} \end{cases} \tag{7.1}$$

The result can be seen in Figure 7.6.

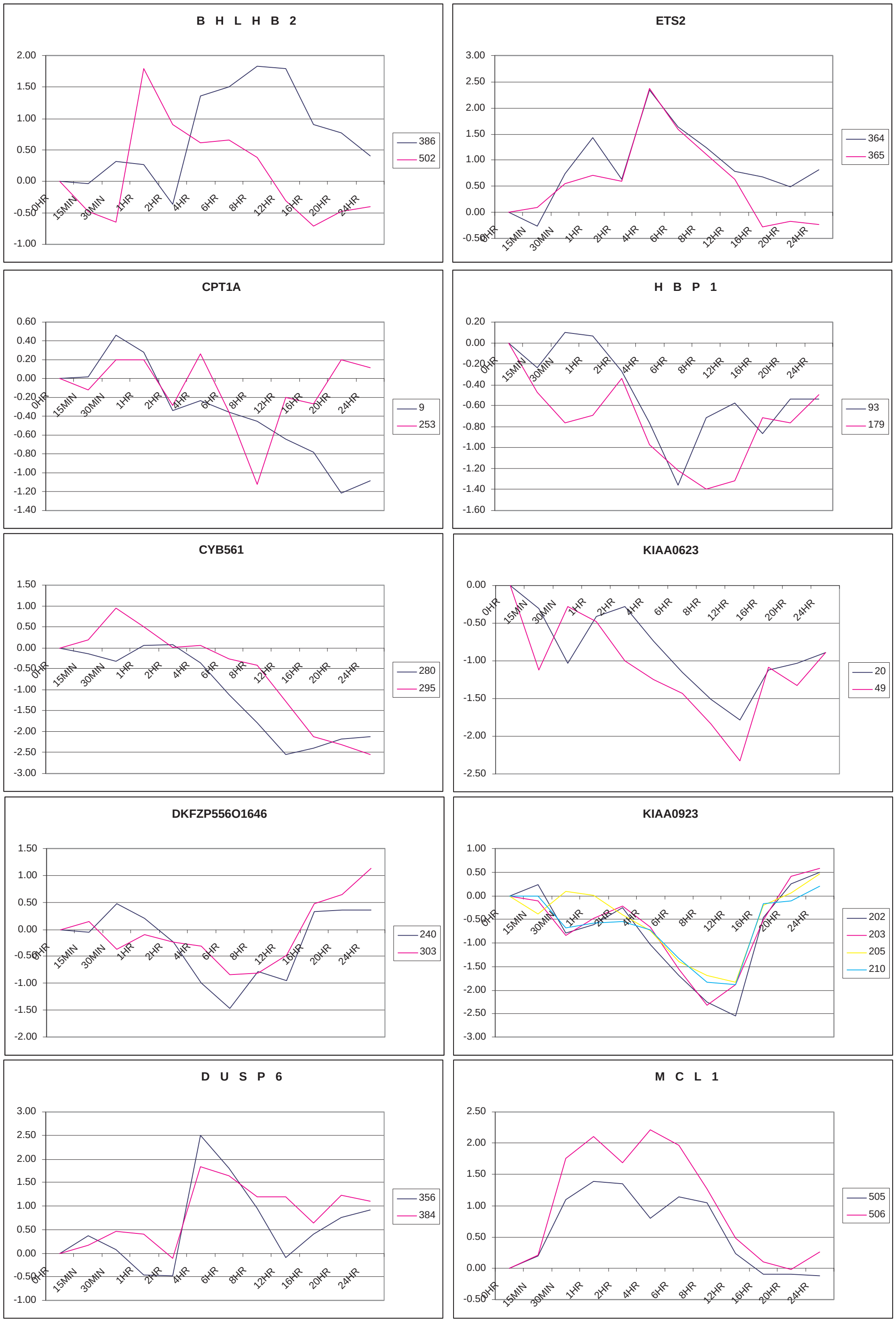
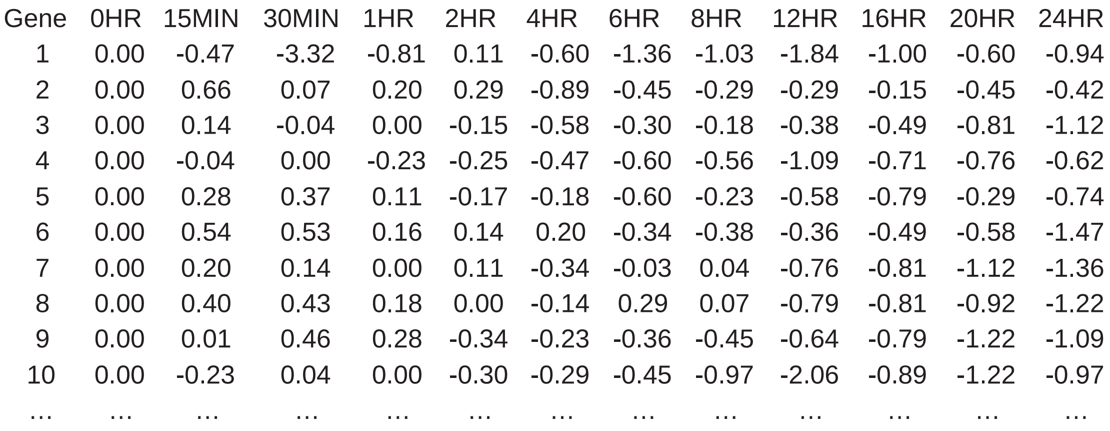


Figure 7.2: The difference in expression level for the same gene measured more than one time. Each set of graphs is marked with the gene's name and each graph is marked with a number. This number indicate how close the the different measurements of the same gene were clustered in [Iyer et al., 1999]. If the numbers are close this indicates that the respective measurements were clustered in the same cluster. If the numbers are far from each other this indicates that the respective measurements were clustered in different clusters.

Indiscernibility relation The Haar Wavelet Transformation and the discretisation enable us to develop an indiscernibility relation that results in a small set of distinguishable clusters by considering only three values (constant, increasing and decreasing) and by requiring stricter similarity for low frequency component than for high frequency components. In more detail, the relation was designed in the following manner; The first attribute in Figure 7.5 was skipped. This attribute represents the constant component of the time series and we were interested in comparing local slopes. For each of the other attributes the difference between the two genes was computed. For the first 7 attributes, no difference was accepted if the two genes should satisfy the indiscernible relation. For the remaining attribute we tolerated a total difference of 3, but not more than 1 for a single attribute.

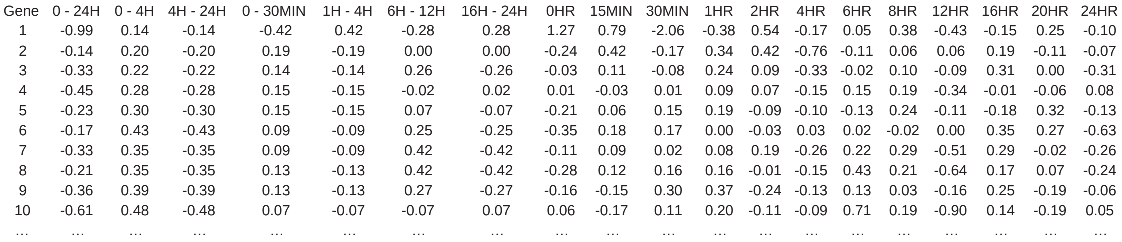
Clustering: The indiscernibility-based clustering algorithm was used on the transformed discrete data, that is, the indiscernibility relation was used on each pair of genes in the set. The result was displayed as an indiscernibility graph where each node represents a gene and two nodes are connected if their respective genes are indiscernible. The graph can be seen in Figure 7.7 and include 473 of the 517 genes in 40 clusters.

Validation: The clusters were validated through inspection by comparing the expression graphs of genes belonging to the same cluster. The expression graphs for genes in the six largest clusters are shown in Figure 7.8. The clusters were also validated against existing knowledge in co-operation with biological experts.



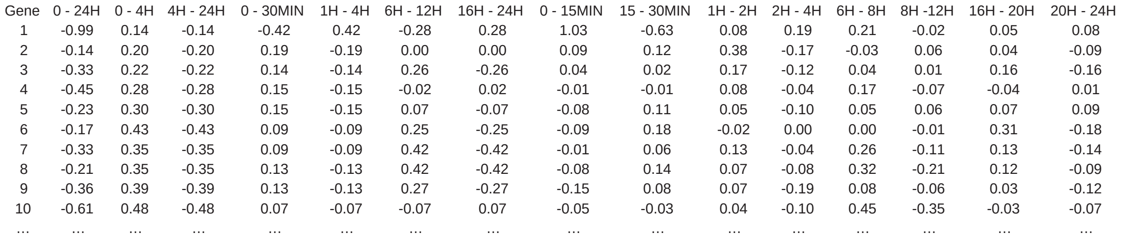
Gene	0HR	15MIN	30MIN	1HR	2HR	4HR	6HR	8HR	12HR	16HR	20HR	24HR
1	0.00	-0.47	-3.32	-0.81	0.11	-0.60	-1.36	-1.03	-1.84	-1.00	-0.60	-0.94
2	0.00	0.66	0.07	0.20	0.29	-0.89	-0.45	-0.29	-0.29	-0.15	-0.45	-0.42
3	0.00	0.14	-0.04	0.00	-0.15	-0.58	-0.30	-0.18	-0.38	-0.49	-0.81	-1.12
4	0.00	-0.04	0.00	-0.23	-0.25	-0.47	-0.60	-0.56	-1.09	-0.71	-0.76	-0.62
5	0.00	0.28	0.37	0.11	-0.17	-0.18	-0.60	-0.23	-0.58	-0.79	-0.29	-0.74
6	0.00	0.54	0.53	0.16	0.14	0.20	-0.34	-0.38	-0.36	-0.49	-0.58	-1.47
7	0.00	0.20	0.14	0.00	0.11	-0.34	-0.03	0.04	-0.76	-0.81	-1.12	-1.36
8	0.00	0.40	0.43	0.18	0.00	-0.14	0.29	0.07	-0.79	-0.81	-0.92	-1.22
9	0.00	0.01	0.46	0.28	-0.34	-0.23	-0.36	-0.45	-0.64	-0.79	-1.22	-1.09
10	0.00	-0.23	0.04	0.00	-0.30	-0.29	-0.45	-0.97	-2.06	-0.89	-1.22	-0.97
…	…	…	…	…	…	…	…	…	…	…	…	…

Figure 7.3: The ten first genes in the Fibroblast data after the logarithmic transformation log_2.



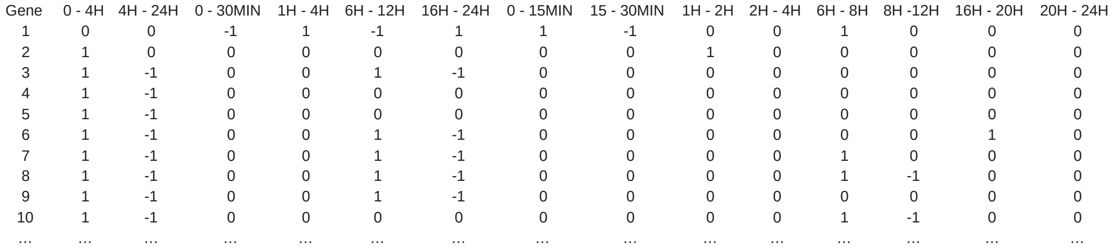
Gene	0 - 24H	0 - 4H	4H - 24H	0 - 30MIN	1H - 4H	6H - 12H	16H - 24H	0HR	15MIN	30MIN	1HR	2HR	4HR	6HR	8HR	12HR	16HR	20HR	24HR
1	-0.99	0.14	-0.14	-0.42	0.42	-0.28	0.28	1.27	0.79	-2.06	-0.38	0.54	-0.17	0.05	0.38	-0.43	-0.15	0.25	-0.10
2	-0.14	0.20	-0.20	0.19	-0.19	0.00	0.00	-0.24	0.42	-0.17	0.34	0.42	-0.76	-0.11	0.06	0.06	0.19	-0.11	-0.07
3	-0.33	0.22	-0.22	0.14	-0.14	0.26	-0.26	-0.03	0.11	-0.08	0.24	0.09	-0.33	-0.02	0.10	-0.09	0.31	0.00	-0.31
4	-0.45	0.28	-0.28	0.15	-0.15	-0.02	0.02	0.01	-0.03	0.01	0.09	0.07	-0.15	0.15	0.19	-0.34	-0.01	-0.06	0.08
5	-0.23	0.30	-0.30	0.15	-0.15	0.07	-0.07	-0.21	0.06	0.15	0.19	-0.09	-0.10	-0.13	0.24	-0.11	-0.18	0.32	-0.13
6	-0.17	0.43	-0.43	0.09	-0.09	0.25	-0.25	-0.35	0.18	0.17	0.00	-0.03	0.03	0.02	-0.02	0.00	0.35	0.27	-0.63
7	-0.33	0.35	-0.35	0.09	-0.09	0.42	-0.42	-0.11	0.09	0.02	0.08	0.19	-0.26	0.22	0.29	-0.51	0.29	-0.02	-0.26
8	-0.21	0.35	-0.35	0.13	-0.13	0.42	-0.42	-0.28	0.12	0.16	0.16	-0.01	-0.15	0.43	0.21	-0.64	0.17	0.07	-0.24
9	-0.36	0.39	-0.39	0.13	-0.13	0.27	-0.27	-0.16	-0.15	0.30	0.37	-0.24	-0.13	0.13	0.03	-0.16	0.25	-0.19	-0.06
10	-0.61	0.48	-0.48	0.07	-0.07	-0.07	0.07	0.06	-0.17	0.11	0.20	-0.11	-0.09	0.71	0.19	-0.90	0.14	-0.19	0.05
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…

Figure 7.4: The ten first genes in the Fibroblast data after the Haar Wavelet Transformation I.



Gene	0 - 24H	0 - 4H	4H - 24H	0 - 30MIN	1H - 4H	6H - 12H	16H - 24H	0 - 15MIN	15 - 30MIN	1H - 2H	2H - 4H	6H - 8H	8H -12H	16H - 20H	20H - 24H
1	-0.99	0.14	-0.14	-0.42	0.42	-0.28	0.28	1.03	-0.63	0.08	0.19	0.21	-0.02	0.05	0.08
2	-0.14	0.20	-0.20	0.19	-0.19	0.00	0.00	0.09	0.12	0.38	-0.17	-0.03	0.06	0.04	-0.09
3	-0.33	0.22	-0.22	0.14	-0.14	0.26	-0.26	0.04	0.02	0.17	-0.12	0.04	0.01	0.16	-0.16
4	-0.45	0.28	-0.28	0.15	-0.15	-0.02	0.02	-0.01	-0.01	0.08	-0.04	0.17	-0.07	-0.04	0.01
5	-0.23	0.30	-0.30	0.15	-0.15	0.07	-0.07	-0.08	0.11	0.05	-0.10	0.05	0.06	0.07	0.09
6	-0.17	0.43	-0.43	0.09	-0.09	0.25	-0.25	-0.09	0.18	-0.02	0.00	0.00	-0.01	0.31	-0.18
7	-0.33	0.35	-0.35	0.09	-0.09	0.42	-0.42	-0.01	0.06	0.13	-0.04	0.26	-0.11	0.13	-0.14
8	-0.21	0.35	-0.35	0.13	-0.13	0.42	-0.42	-0.08	0.14	0.07	-0.08	0.32	-0.21	0.12	-0.09
9	-0.36	0.39	-0.39	0.13	-0.13	0.27	-0.27	-0.15	0.08	0.07	-0.19	0.08	-0.06	0.03	-0.12
10	-0.61	0.48	-0.48	0.07	-0.07	-0.07	0.07	-0.05	-0.03	0.04	-0.10	0.45	-0.35	-0.03	-0.07
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Figure 7.5: The ten first genes in the Fibroblast data after the Haar Wavelet Transformation II.



Gene	0 - 4H	4H - 24H	0 - 30MIN	1H - 4H	6H - 12H	16H - 24H	0 - 15MIN	15 - 30MIN	1H - 2H	2H - 4H	6H - 8H	8H -12H	16H - 20H	20H - 24H
1	0	0	-1	1	-1	1	1	-1	0	0	1	0	0	0
2	1	0	0	0	0	0	0	0	1	0	0	0	0	0
3	1	-1	0	0	1	-1	0	0	0	0	0	0	0	0
4	1	-1	0	0	0	0	0	0	0	0	0	0	0	0
5	1	-1	0	0	0	0	0	0	0	0	0	0	0	0
6	1	-1	0	0	1	-1	0	0	0	0	0	0	1	0
7	1	-1	0	0	1	-1	0	0	0	0	1	0	0	0
8	1	-1	0	0	1	-1	0	0	0	0	1	-1	0	0
9	1	-1	0	0	1	-1	0	0	0	0	0	0	0	0
10	1	-1	0	0	0	0	0	0	0	0	1	-1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Figure 7.6: The ten first genes in the Fibroblast data after discretisation.

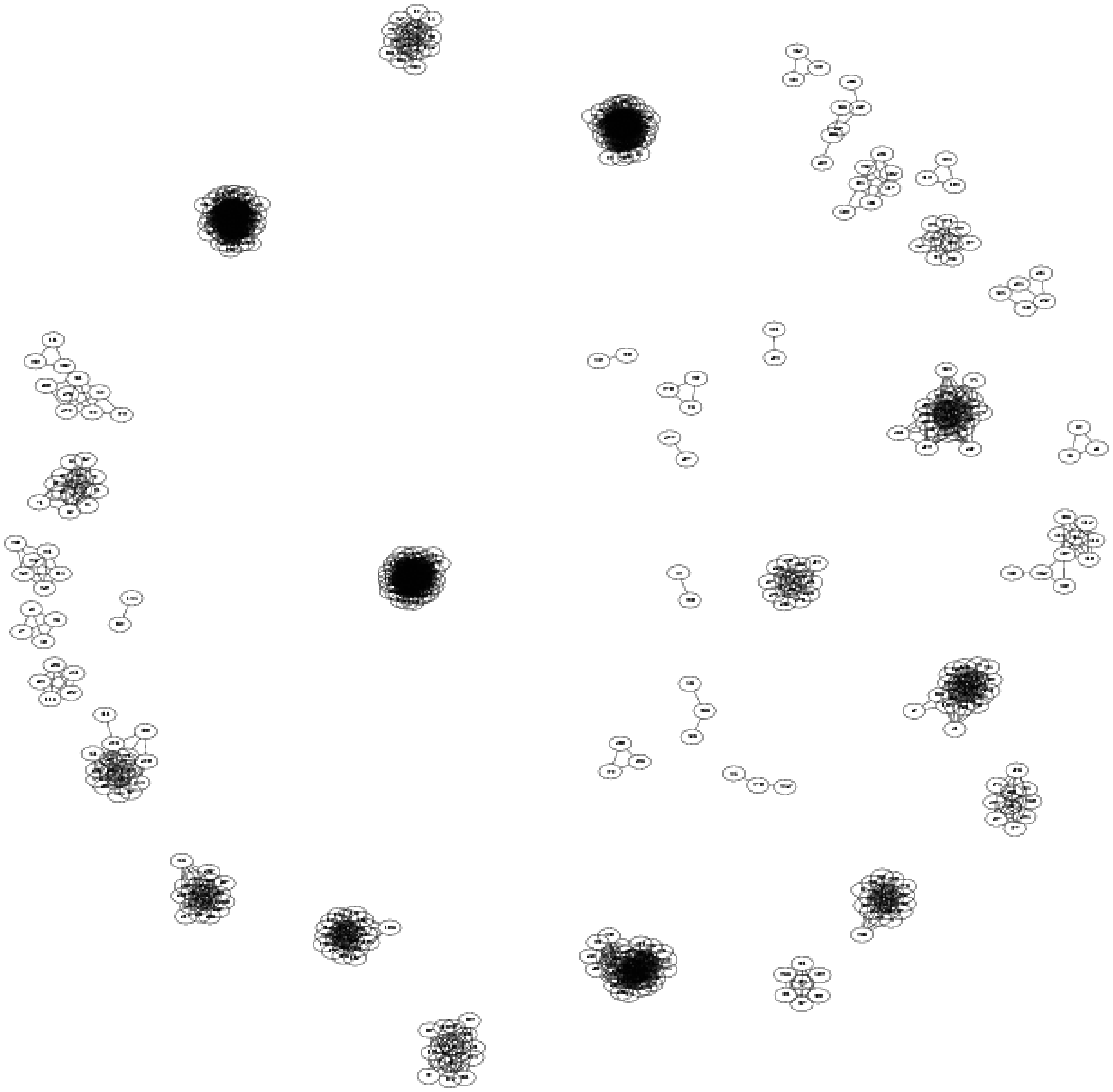


Figure 7.7: The indiscernibility graph showing the resulting clusters from applying the indiscernibility-based clustering approach to the Fibroblast data.

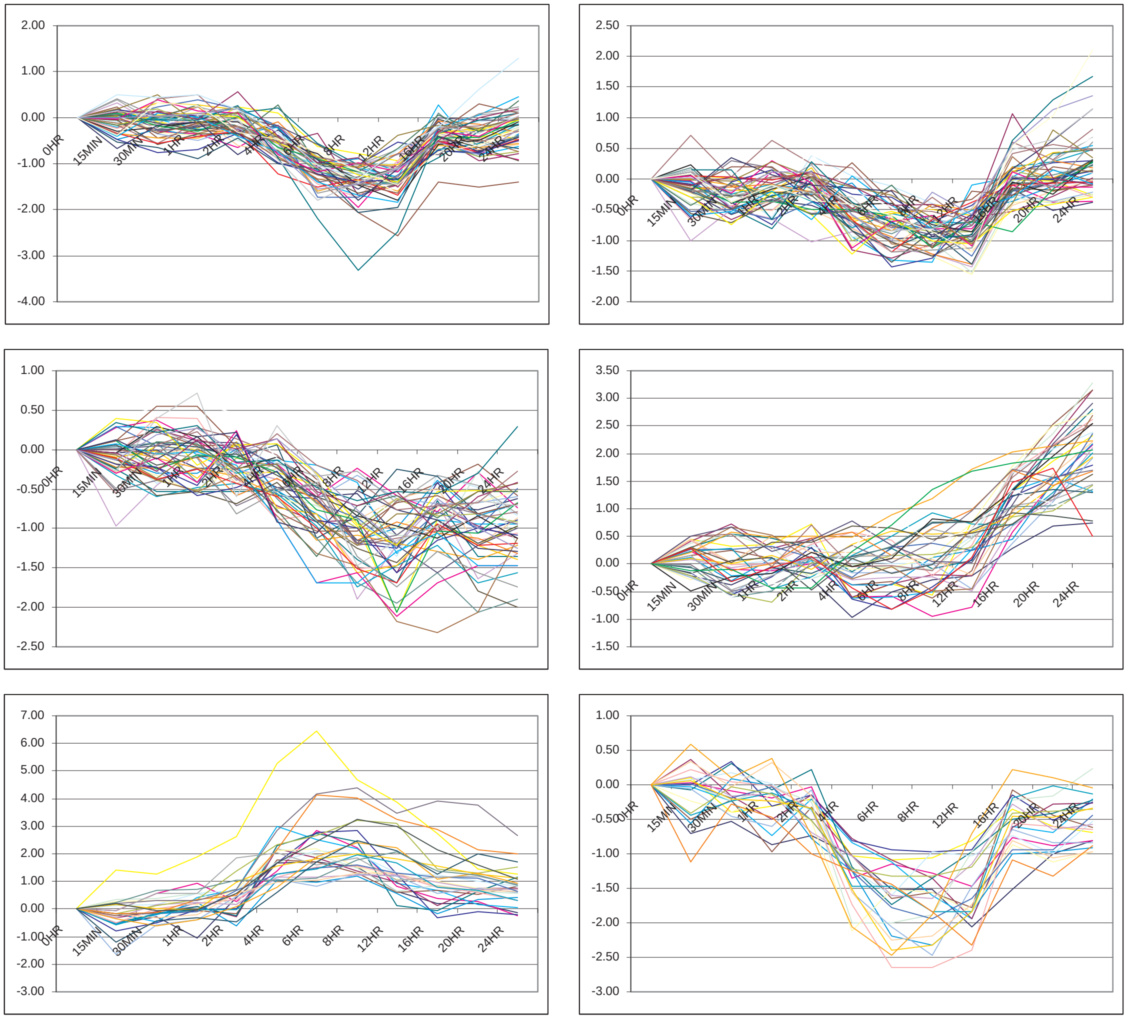


Figure 7.8: Expression level graphs showing the similarity between genes belonging to the same cluster. The six largest clusters from Figure 7.7 is shown.

The results outlined above show have the indiscernibility-based clustering used together with discretisation and the Haar Wavelet Transformation is capable of mapping a large part of the genes in the Fibroblast data into a few clusters. In fact, the six largest clusters contain 50% of the genes. The expression graphs in Figure 7.8 show how the genes within the same cluster reflect similar curve properties. The similarity within a cluster is of course a result of the strictness of the indiscernibility relation. Our goal in this analysis was to group the genes into a few large clusters. The result, as one might expect, is a number of small clusters accompanying the larger ones. This is due to the fact that the natural underlying groups in the data set vary both in size and tightness.

The clusters in Figure 7.7 show a clear similarity to the clusters obtained and presented in [Iyer et al., 1999]. 80% of the genes in the three largest cluster (corresponding to the three first clusters in Figure 7.8 going from left to right) belong to the same clusters in [Iyer et al., 1999]. The remaining 20% belong to one or at most two additional clusters.

Our bio-medical experts (Astrid Lægreid and Arne Sandvik) found the results interesting. We were able to present an overview showing what kind of gene expression variations that were present in the data set and which genes following which type of variation. However, as the biologists sat down to validate the clusters against the existing knowledge, a number of questions and new challenges appeared. Basically, the problem was two-folded:

1. A lot of information escapes the clusters due to the requirement of similarity over the full 24 hour period. Information about genes showing a similar expression profile in shorter periods is not captured.
2. The expression profiles for a cluster do very rarely follow the typical idealised profile for a gene executing a particular function. This makes the results difficult to interpret.

A solution to problem number one is simply to investigate sub-intervals of the time series. As a result we should capture similarity in shorter periods, and thus gain knowledge about genes which partly execute the same function and partly execute different functions. This approach will result in overlapping clusters, that is, one gene occurs in more than one cluster.

A solution to the second problem seems to be to define idealised profiles in advance and then group the genes according to their similarity to these profiles. In the next section these two solutions will be merged into a template-based analysis of sub-intervals.

Before leaving the indiscernibility-based clustering approach we look into alternative interpretations of indiscernibility. Figure 7.9 shows the indiscernibility graph resulting from designing the relation in such a way that two genes satisfy it if they have mirrored expression profiles. The graph was obtained using the same clustering method as before except for the definition of the relation. Figure 7.10 shows expression graphs for genes with mirrored expression profiles grouped in four typical clusters. This proves that *related* genes not necessarily need to be *similar*. There is a lot of possibilities when it comes to defining (biologically) relevant relations between two genes. Finding genes with causally related expression profiles would for example be (another) natural extension of the analysis described above.

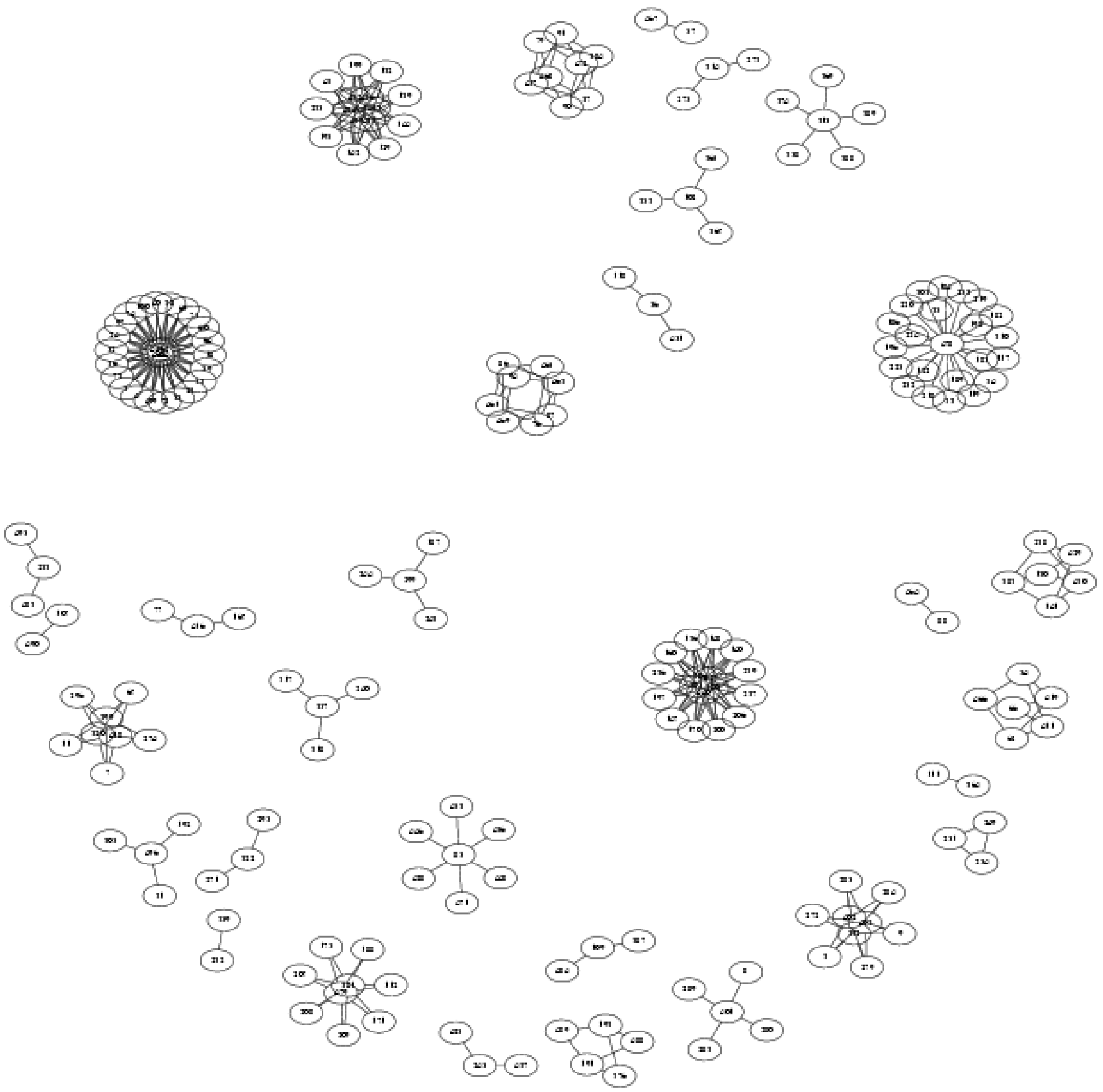


Figure 7.9: The indiscernibility graph of the clusters resulting from defining related genes as being genes with mirrored expression profiles.

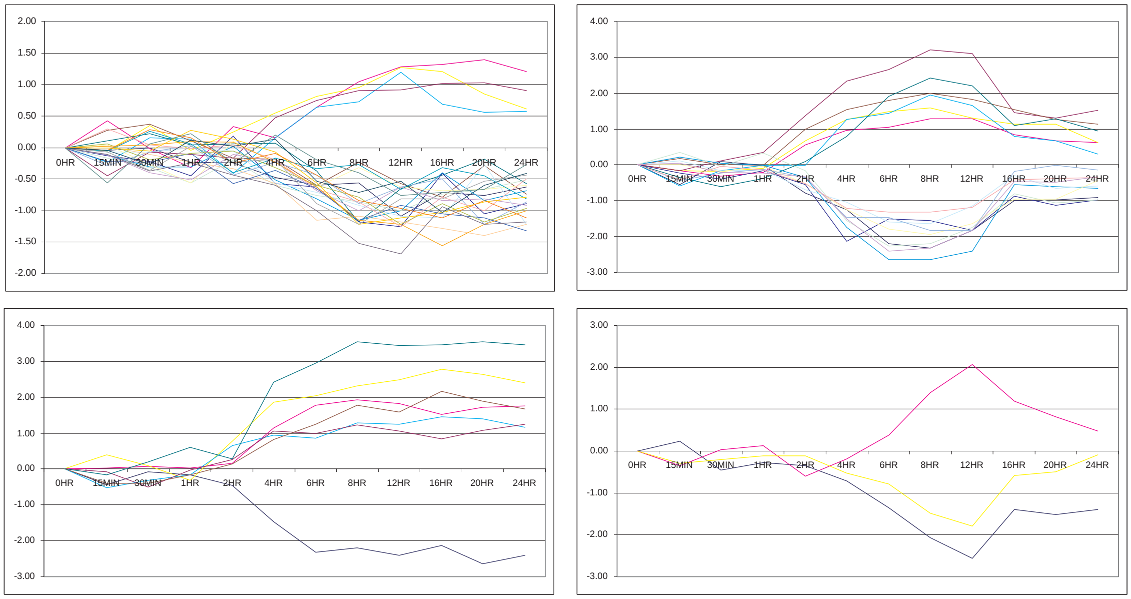


Figure 7.10: Four typical clusters containing genes with mirrored expression profiles.

7.3 Approach 2: Template-based Clustering Analysis

This section describes the approach of applying the template-based clustering method from Chapter 6 to the Fibroblast data. This was done in order to solve some of the problems revealed using the indiscernibility-based method presented in the last section. The following method was used:

Templates A set of templates, or idealised expression profiles, were defined that reflect the basic reactions of a gene to serum (or, in general, to any change in the environment). The following five templates were defined:

- Constant: A gene matches the *constant*-template in a given interval if its expression level does not vary more than 0.2 from the mean expression level in this interval.
- Increasing: A gene matches the *increasing*-template in a given interval if its expression level increases by more than one over the full interval and the increase fulfil the following requirements: (1) The increase during the first and last “atomic” interval[1] in the given full interval most exceed 0.2 and the expression level should never drop below / above this level later. (2) The expression level is allowed to decrease during the interval, but never by more than 0.2 during an “atomic” interval.
- Decreasing: The *decreasing*-template is defined in the similar manner as the *increasing*-template.
- Increasing-decreasing: A gene matches the *increasing-decreasing*-template in a given interval if the interval can be divided into two sub-intervals such that the

[1] By an “atomic” interval we mean the interval between a given time point and the nearest following time point.

gene matches the *increasing*-template in the first interval and the *decreasing*-template in the second interval. However, the sub-matches do not have to satisfy the end- and start-requirements of an increase, respectively decrease, of 0.2.

- Decreasing-increasing: The *decreasing-increasing*-template is defined in the similar manner as the *increasing-decreasing*-template.

Matching table A *matching table* was defined such that each gene represent a row and each possible sub-interval represent a column. The entry in this table is the template that the given gene matches in the given interval. Matches for the *increasing*- and *decreasing*-templates most be defined over at least three time points. The three other templates most be defined over at least four time points.

Simplification The matching table was simplified in that all redundant matches were removed. That is, all matches occurring fully within a sub-interval of another match were removed.

Ordering The intervals were ordered according to how many genes they were describing. This was done once for each of the five templates so we ended up with five ordered sets of intervals.

Validation Biological experts were given the ordered sets of intervals and the genes contained in the clusters determined by an interval and its corresponding template. Since all matches in all sub-intervals are considered, we can clearly call a cluster determined by a given interval and a given template an *atomic cluster*. The biological experts would be looking for a set of clusters that describe the dynamics of the Fibroblast data. These clusters would hence be composed of a set of atomic clusters or be atomic clusters themselves. The largest clusters for each of the five templates are presented as expression graphs in Figure 7.11.

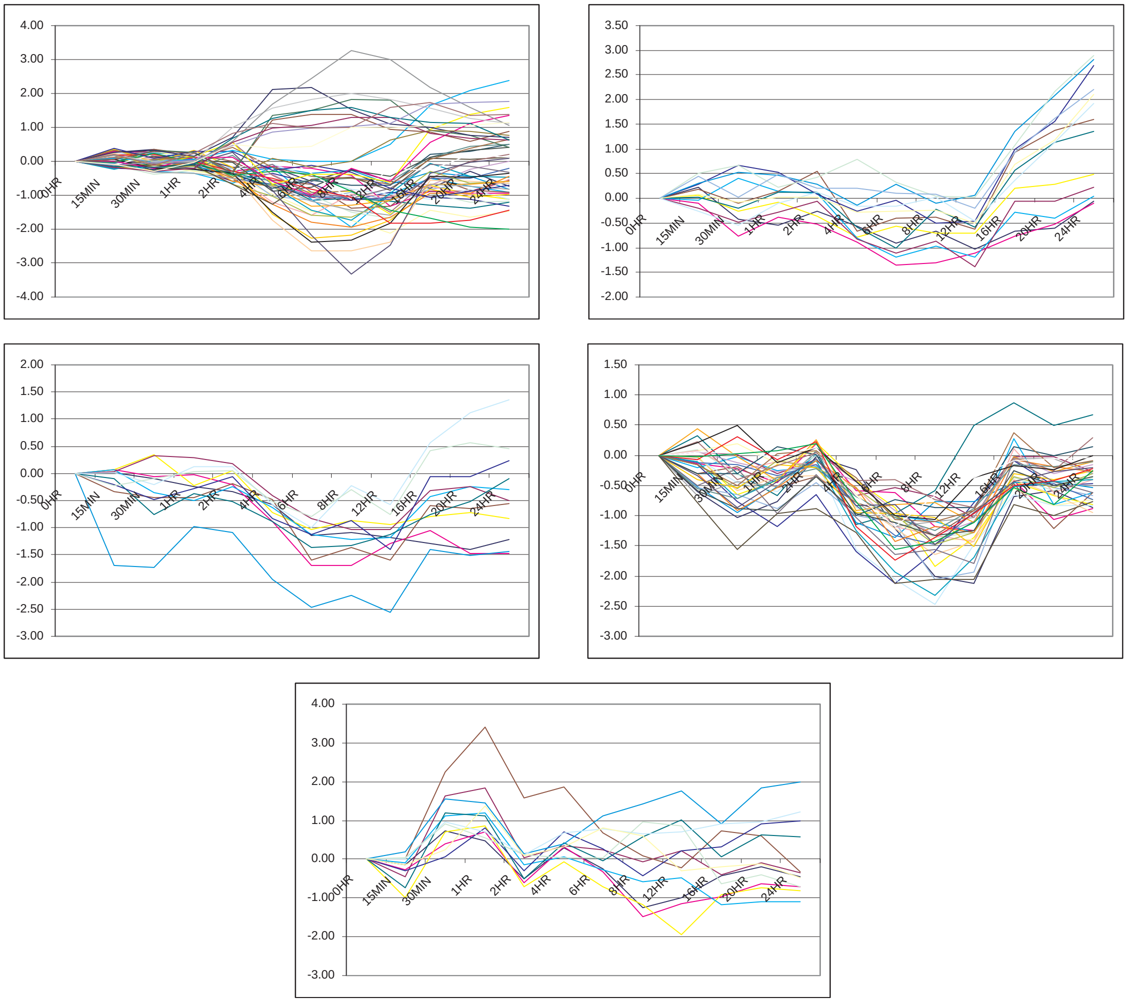


Figure 7.11: The largest clusters for each of the five templates presented as expression graphs. The graphs show, from left to right, (1) genes matching the *constant-template* from 0 min. to 1H, (2) genes matching the *increasing-template* from 12H to 24H, (3) genes matching the *decreasing-template* from 2H to 6H, (4) genes matching the *decreasing-increasing-template* from 2H to 16H and (5) genes matching the *increasing-decreasing-template* from 15 min. to 2H.

7.4 Biological Interpretation

Having defined the atomic clusters as described, we can now turned our attention to the biological interpretation. This includes (1) finding a set of super-clusters describing the data set, (2) validating these clusters against existing knowledge, that is, investigate whether the known genes in the data set are clustered in agreement with their function and (3) describing the different genes' function on the basis of (1) and (2) and in particular the function of unknown genes.

The first step was handled using biological knowledge about the different phases of genes responding to some environmental change. A set of super-intervals were used to cluster the atomic clusters of each template into super-clusters. The super-intervals used were

Immediate early: From 0 to 1 hour.

Delayed immediate early: From 1 to 4 hours.

Intermediate: From 4 hours.

Late: From 8 hours.

The second step includes both the collection of knowledge about the known genes in the data set and the comparison of this knowledge to our obtained clusters. The collection of existing knowledge was handled partly manually and partly automatic from the Ashburner gene ontology ([Ashburner, 1999]). This ontology is thorough discussed in [Tjeldvoll, 1999]. Having placed all known genes in the ontology, the clusters can easily be validated by investigating how near each-other genes in the same cluster are situated in the ontology. In the Fibroblast data we selected seven different processes in which a gene can be involved, and then defined two genes to have “similar functions” if they appeared in the same process-sub-tree in the ontology. The processes were

- Transcription
- Protein synthesis
- Stress response
- Lipid synthesis
- Organelle biogenesis
- Cell motility
- Re-entry cell cycle

The collection of existing knowledge is still not finished and this reduces our ability to draw any definitive conclusions. However, the preliminary results are promising and show a clear mapping between our obtained clusters and the processes listed above.

The third step is difficult since the second step is not yet completed . However, we will give some general remarks about the biological interpretation of our clusters here. Our analysis shows a large degree of dynamics in the Fibroblast data. Even after removing redundant matches a large number of intervals are needed in order to capture all gene profiles matching the different templates. The table below shows how many genes matched the different templates, how many times they matched it and how many intervals which were needed in order to capture all these matches (the number in parenthesises indicates how many of the intervals that only contained one match):

Template	Genes matching	Total matches	Intervals needed
Constant	195	248	24(7)
Increasing	105	109	38(12)
Decreasing	122	127	39(15)
Increasing - Decreasing	143	156	39(9)
Decreasing - Increasing	211	225	37(8)

The analysis shows that genes matching the same template in a given interval also have a clear tendency of expressing similar profiles over the full 24 hour period. However, we also find clear examples of this not being the case. Hence our hypothesis about loosing significant information when requiring similarity over the full 24 hour period seems confirmed.

During our analysis of the Fibroblast data we have discovered a number of problems that make the analysis of gene expression data a big challenge. One problem is the complexity of biological systems in general. This naturally results in large data sets and complex dependencies between both objects and attributes in the data set. Also the interpretation of computer generated results is a big challenge. Take for example the interrelationship between a gene's expression level and its corresponding protein. First of all there is a delay between when a gene starts being expressed, which is what we measure in terms of mRNA level, and when the protein starts to act, which is our biological interpretation. Further this delay takes on different lengths from gene to gene. Also a protein's time-profile (how the amount of protein vary with time) does not necessarily follow the time-profile of its mRNA. This is just some of the aspects one needs to take into consideration when analysing gene expression data.

Chapter 8

Tumour and Normal Colon Tissues

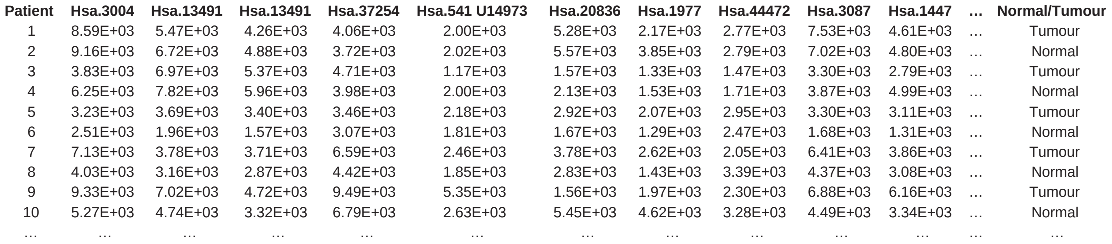
In this chapter we will present a clustering analysis of the “tumour and normal colon tissue” data set described in [Alon et al., 1999]. Moreover, we will use the rough set framework described in Chapter 4 to induce a model that recognises the clusters. Two clustering algorithms taken from the example in Chapter 5 (The Party Families in Europe 1970-1992) will be applied on the data:

1. The Ward’s hierarchical clustering algorithm implemented as an agglomerative method, representing the hierarchical clustering methods.
2. The k-means algorithm, representing the converging non-hierarchical clustering methods.

8.1 The Data Set

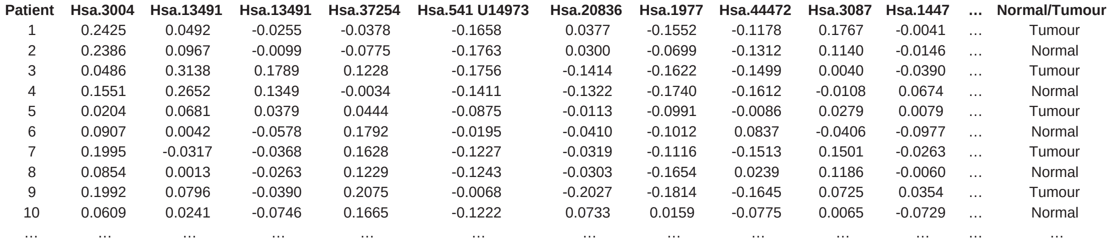
[Alon et al., 1999] investigates the expression profile of over 6500 genes in 40 tumour and 22 normal colon tissue samples (this data set is publicly available on the WEB: < http : //www.molbio.princeton.edu/colondata >). A subset of 2000 genes with the highest minimal intensity across the samples was analysed. The data was clustered both with respect to genes and with respect to tissue. The method used was the one of organising the data set into a binary tree where genes are near each other on the ”gene tree” if they show a strong correlation across experiments, and tissues are near each other on the ”tissue tree” if they have similar gene expression profiles. The binary tree was constructed by using a divisive implementation of the hierarchical clustering method.

We will use two methods to cluster the data set with respect to tissue. In this way we can easily measure and compare the results by observing how well the different methods distinguish tumour tissue from normal tissue. We can also induce the model over the known partition. Investigating the data with respect to tissue gives us an information system with 62 objects (patients) and 2000 attributes (genes). A segment of the data is shown in Figure 8.1. Note that this data set is not time series as analysed in the last chapter. Thus each patient represents a point in the 2000-dimensional vector space. The values were normalised such that the sum of the components of each vector was zero and the magnitude was one. A segment of the normalised data can be seen in figure 8.2. The results of the analysis are outlined below.



Patient	Hsa.3004	Hsa.13491	Hsa.13491	Hsa.37254	Hsa.541 U14973	Hsa.20836	Hsa.1977	Hsa.44472	Hsa.3087	Hsa.1447	...	Normal/Tumour
1	8.59E+03	5.47E+03	4.26E+03	4.06E+03	2.00E+03	5.28E+03	2.17E+03	2.77E+03	7.53E+03	4.61E+03	...	Tumour
2	9.16E+03	6.72E+03	4.88E+03	3.72E+03	2.02E+03	5.57E+03	3.85E+03	2.79E+03	7.02E+03	4.80E+03	...	Normal
3	3.83E+03	6.97E+03	5.37E+03	4.71E+03	1.17E+03	1.57E+03	1.33E+03	1.47E+03	3.30E+03	2.79E+03	...	Tumour
4	6.25E+03	7.82E+03	5.96E+03	3.98E+03	2.00E+03	2.13E+03	1.53E+03	1.71E+03	3.87E+03	4.99E+03	...	Normal
5	3.23E+03	3.69E+03	3.40E+03	3.46E+03	2.18E+03	2.92E+03	2.07E+03	2.95E+03	3.30E+03	3.11E+03	...	Tumour
6	2.51E+03	1.96E+03	1.57E+03	3.07E+03	1.81E+03	1.67E+03	1.29E+03	2.47E+03	1.68E+03	1.31E+03	...	Normal
7	7.13E+03	3.78E+03	3.71E+03	6.59E+03	2.46E+03	3.78E+03	2.62E+03	2.05E+03	6.41E+03	3.86E+03	...	Tumour
8	4.03E+03	3.16E+03	2.87E+03	4.42E+03	1.85E+03	2.83E+03	1.43E+03	3.39E+03	4.37E+03	3.08E+03	...	Normal
9	9.33E+03	7.02E+03	4.72E+03	9.49E+03	5.35E+03	1.56E+03	1.97E+03	2.30E+03	6.88E+03	6.16E+03	...	Tumour
10	5.27E+03	4.74E+03	3.32E+03	6.79E+03	2.63E+03	5.45E+03	4.62E+03	3.28E+03	4.49E+03	3.34E+03	...	Normal
...	...	...	...	...	...	...	...	...	...	...	...	...

Figure 8.1: The ten first genes and the ten first patients in the original tumour and normal tissue data set.



Patient	Hsa.3004	Hsa.13491	Hsa.13491	Hsa.37254	Hsa.541 U14973	Hsa.20836	Hsa.1977	Hsa.44472	Hsa.3087	Hsa.1447	...	Normal/Tumour
1	0.2425	0.0492	-0.0255	-0.0378	-0.1658	0.0377	-0.1552	-0.1178	0.1767	-0.0041	...	Tumour
2	0.2386	0.0967	-0.0099	-0.0775	-0.1763	0.0300	-0.0699	-0.1312	0.1140	-0.0146	...	Normal
3	0.0486	0.3138	0.1789	0.1228	-0.1756	-0.1414	-0.1622	-0.1499	0.0040	-0.0390	...	Tumour
4	0.1551	0.2652	0.1349	-0.0034	-0.1411	-0.1322	-0.1740	-0.1612	-0.0108	0.0674	...	Normal
5	0.0204	0.0681	0.0379	0.0444	-0.0875	-0.0113	-0.0991	-0.0086	0.0279	0.0079	...	Tumour
6	0.0907	0.0042	-0.0578	0.1792	-0.0195	-0.0410	-0.1012	0.0837	-0.0406	-0.0977	...	Normal
7	0.1995	-0.0317	-0.0368	0.1628	-0.1227	-0.0319	-0.1116	-0.1513	0.1501	-0.0263	...	Tumour
8	0.0854	0.0013	-0.0263	0.1229	-0.1243	-0.0303	-0.1654	0.0239	0.1186	-0.0060	...	Normal
9	0.1992	0.0796	-0.0390	0.2075	-0.0068	-0.2027	-0.1814	-0.1645	0.0725	0.0354	...	Tumour
10	0.0609	0.0241	-0.0746	0.1665	-0.1222	0.0733	0.0159	-0.0775	0.0065	-0.0729	...	Normal
...	...	...	...	...	...	...	...	...	...	...	...	...

Figure 8.2: The ten first genes and the ten first patients in the tumour and normal tissue data set after normalisation.

8.2 The Ward's hierarchical clustering algorithm

The Ward's hierarchical clustering algorithm was implemented using the SSE (Sum of Square Error) as the quality measure. Thus the two clusters resulting in the smallest increase in SSE value were successively merged. Figure 8.3 shows the SSE value and the average SSE value over the clusters as a function of the number of clusters. These graphs can often be used as a help to choose a stop criterion for the algorithm, although in this example no clear break point appears. Because of this we inspected how the algorithm distinguished normal tissue from tumour tissue by gradually reducing the number of clusters. The result can be seen in Table 8.1. From this table we see how two clusters are successively merged. Clearly, four clusters give us a rather good partitioning which distinguish normal tissue from tumour tissue.

No. of Clusters	*Normal:Tumour*
2	15:14 , 7:26
3	15:14 , 3:11 , 4:15
4	11:3 , 4:11 , 3:11 , 4:15
5	11:3 , 4:11 , 3:11 , 2:6 , 2:9

Table 8.1: The table shows how the Ward's hierarchical clustering method distinguishes normal tissue from tumour tissue for different numbers of clusters. Each of the *normal:tumour*-pairs indicate how many normal tissues and how many tumour tissues there were in this cluster.

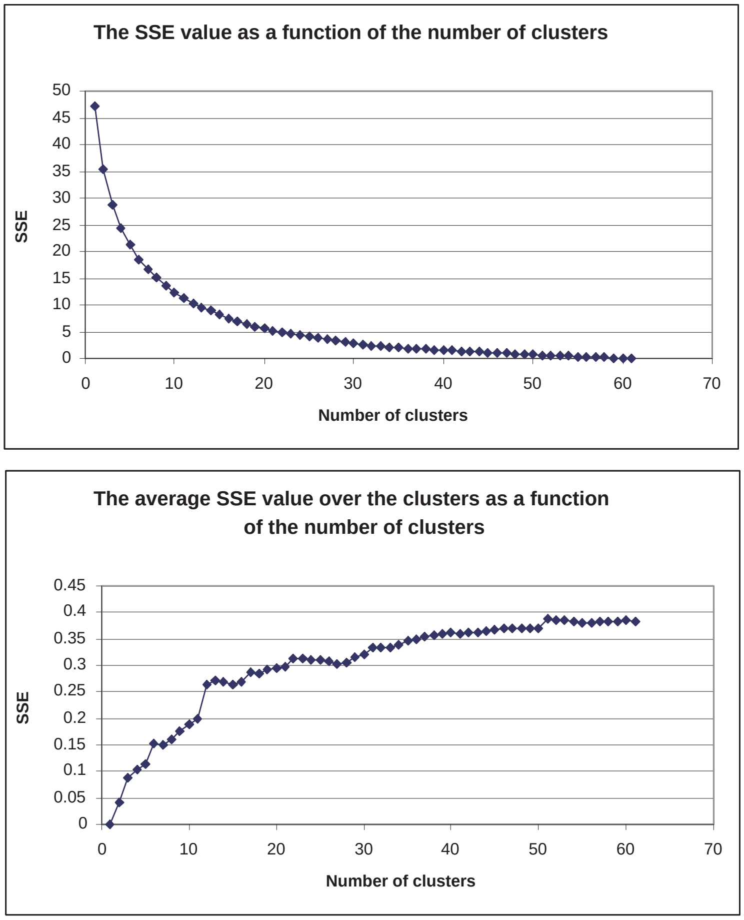


Figure 8.3: The SSE value and the average SSE value over the clusters as a function of the number of clusters.

8.3 The k-means algorithm

The k-means algorithm was implemented using the Euclidean distance as a similarity measure. The data set was initially divided into 4 clusters. The result of four different initial assignments of objects into clusters can be seen in Table 8.2.

Configuration	*Normal:Tumour*
4a	13:1 , 1:20 , 1:6 , 7:13
4b	1:8 , 7:2 , 12:3 , 2:27
4c	2:8 , 5:10 , 13:3 , 2:19
4d	1:6 , 14:3 , 3:15 , 4:16

Table 8.2: The table shows how the k-means clustering method distinguishes normal tissue from tumour tissue for different initial configuration using 4 clusters. Each of the *normal:tumour*-pairs indicate how many normal tissues and how many tumour tissues there were in this cluster.

Again we see that the results from the k-means algorithm strongly depend on the initial configuration. But nonetheless it seems to distinguish normal tissue from tumour tissue relatively well. It is interesting to notice that in order to distinguish the two classes (tumour and normal tissue) we need to divide the data set into more than two clusters. Using only two cluster really did not work at all. This indicates that the tumour and the normal tissue classes themselves contain sub-clusters and that some of these sub-clusters lie closer, in this case in terms of Euclidean distance in the 2000-dimensional space, to sub-clusters in the other class than to sub-clusters in its own class.

The results from clustering tissue in [Alon et al., 1999] show some of the same problems as we discussed above. Neither that work is capable of perfectly distinguishing normal tissue from tumour tissue, that is, some tumour tissues appear in the “normal tissue”-clusters and some normal tissues appear in the “tumour tissue”-clusters.

Concluding that we are capable of distinguishing tumour from normal tissue, we now turn our attention to inducing a classifier (supervised learning) that recognises the two classes. Of course, normally we would not know which objects belong to which class and would thus have to trust our clustering algorithms. However, biological experts can use their knowledge to modify, or fine-tune, the clusters, and we will use this as an explanation to induce the model on the basis of the original partition of the data set.

8.4 Classification: Inducing a Model

Using the rough set framework to induce propositional rules as explained in Chapter 4, we can obtain a rule set that can classify tumour tissue and normal tissue. We used the 10-fold cross validation approach to split the data set into training sets and test sets. The data was discretised using the Boolean approach in that the minimal set of cuts that does not loose knowledge needed to discern tumour tissue from normal tissue is found (sse [Nguyen and Skowron, 1995]. Then the Johnson algorithm for reduct computation was used (see [Johnson, 1974]). This algorithm finds one reduct for each object in linear time. This reduct is capable of discerning this object from all other objects not belonging to the same class. The whole modelling process is supported by the ROSETTA system.

The 10-fold cross validation method split the data set into ten sub-sets such that one of the sub-sets act as a test set and the union of the nine other sub-sets act as a training set. This procedure is repeated ten times such that all sub-sets act as a training set once. The results can be seen in Table 8.3.

The results show that we are capable of inducing a model that recognises the two classes relatively well. Remember that the clustering algorithms gave us more than two clusters and that we concluded that the two classes of normal and tumour tissues themselves must consist of underlying sub-clusters. As a result, the classifier needs to distinguish four or five classes rather than two. Given the fact that we have as few as 62 objects altogether, that is 56 objects in the training set and 6 objects in the test set, this might explain why the results in terms of accuracy and AUC are not as good as we would have hoped for. Also the large number of attributes (genes) increases the complexity of the modelling task.

Given new cases with possible tumour we can now use the classifier to diagnose them. Moreover, the rules in the classifier, or more correctly the reducts on which the rules are based, can be used to discover knowledge about which genes are responsible for tumour and which are not. We will not go into the biological interpretation of this specific data set. Instead we will stress the usefulness of doing supervised learning on the result of unsupervised learning as exemplified in this chapter.

Note that the case study in this chapter does not address the problem of mapping genes to functions in the same way as explained in Chapter 3. To do that we would need to cluster genes according to their expression level and not tissues according to the measured genes' expression level.

Split	Accuracy	AUC
1	0.67	0.67
2	0.83	1.00
3	1.00	0.50
4	0.33	0.44
5	0.83	1.00
6	0.83	1.00
7	0.50	0.67
8	0.83	0.40
9	0.50	0.55
10	0.88	1.00
Average	0.72	0.72
Standard deviation	0.21	0.25

Table 8.3: The table shows the quality of the classifier induced by the ROSETTA system.

Part IV

Evaluation of the Methodology

This part discusses the quality of our proposed methodologies and our experiences applying them to real world gene expression data sets. It further summarises and concludes the work done in this thesis. Finally, it describes the work that still needs to be done.

Chapter 9

Discussion, Conclusions and Further Work

In this chapter we will discuss our results and experiences and try to draw our work to a conclusion.

9.1 Discussion

9.1.1 Syntactical Clustering Methods

The most difficult decision when using the syntactical methods is to select the right number of clusters. This was also discussed in Chapter 5. Although the hierarchical methods investigate all possible number of clusters we still need to select one of them. Using a stop criterion is not easy unless a natural choice emerges from some domain specific interpretation. Plotting the SSE value as a function of the number of clusters like we did in our analysis of the "tumour and normal tissue" analysis in chapter 8 might be helpful. However, if no break in the curve appears, like in Figure 8.3, this is of no help. One should remember that the "optimal" number of cluster given a data set is relative both to the choice of similarity measure and to the intentions of the people who analyses the data. An evaluation of all factors including domain knowledge is often needed to select the right number of clusters.

The non-deterministic nature of the non-hierarchical clustering algorithms, like the k-means algorithm, can be a problem. However, the fact that the resulting clusters change with different initial configurations might be used to select the number of initial clusters where the final clusters are relatively stable.

The indiscernibility-based clustering approach somewhat avoids the problem of selecting a number of clusters since this number comes as a result of the definition of the indiscernibility relation. Hence, we can adjust the relation a sufficient number of objects is grouped in a favourable number of clusters. The biggest challenge using this method is to design the indiscernibility relation such that the algorithm results in distinguishable clusters. Often we either cluster very few objects or all objects are related in one big cluster. We solved this problem in the time series domain by applying the Haar Wavelet Transformation and discretisation. However, in vector space this problem might be even more difficult to solve.

9.1.2 Semantical Clustering Methods

The introduction of domain knowledge into the KDD process is often important, but also difficult. In the clustering process, domain knowledge can be introduced at three different levels:

Similarity measure: Domain knowledge can be introduced in the similarity measure by defining it in a biological relevant way. This can be done by for example weighting the different attributes or even introducing new attribute through a transformation. The Boolean similarity measures might be more suited for the introduction of domain knowledge than, for example, distance measures, since they do not have to be continuous functions. They might therefore be easier to design with respect to knowledge.

Number of clusters: Domain knowledge can be used to select the resulting number of clusters. For non-hierarchical algorithms, where an initial configuration of clusters is selected, domain knowledge might even be used to select a favourable initial configuration. That is, biological knowledge is used to initially assign the genes to clusters or to select the seed points. Thus the clustering algorithm just fine-tunes this initial configuration.

Templates: The biologically relevant features characterising each cluster can be predefined and the objects distributed according to how they match these features.

Introducing domain knowledge in the similarity measure and in selecting the number of initial clusters can be done also in syntactical clustering. It is when the features of each cluster are predefined and the clustering algorithm is reduced to pattern recognition or template matching that we call it semantical clustering or knowledge-based clustering.

The syntactical approaches have clear advantages in that they find the existing patterns in the data without any predefined assumptions. However, when working with domain experts they often want to test a specific hypothesis. This can be done more explicitly be using the hypothesis to define templates rather than using the result of a semantical clustering algorithm to verify the hypothesis afterwards. We believe that syntactical and semantical clustering should be viewed as complementary methods rather than competing methods. The syntactical methods could be used for initial analysis in order to reveal the patterns hidden in the data. The semantical methods could then be used to test more advanced hypotheses based on the initial analysis.

9.1.3 Validating clusters

Clusters in the time series domain are in many aspects easy to validate compared to clusters in the vector space domain. Comparing the different objects as a function of time easily helps us validate whether objects in the same cluster really reflect the similarity we are looking for. In vector space, drawing dendrogram can help us obtain knowledge about similar objects by following the evolution of the algorithm as it merges or splits clusters. Also, the indiscernibility graphs gives us extra information about the relation between objects in the same cluster. [Johnson and Wichern, 1998] discuss methods of transforming multivariate data into a low-dimensional space in order to display it preferably in two or three dimensions.

When working with gene expression data and biological knowledge, the portion of known genes in the data set can be used to validate the clusters. If all, or a large part of, the known genes in one cluster are known to have the same biological function, this might indicate that the cluster are good. However, one should always take account for the distribution of known genes in the data set. If genes responsible for some specific biological function are over-represented in the data set, the fact that a large part of one cluster consist of these genes does not have to mean that this cluster is biologically relevant. The statistical significance of the results should always be investigated carefully.

9.1.4 Supervised vs. Unsupervised Learning

In the beginning of this thesis we stated that our primary goal was to design and use computational methods for finding the function of unknown genes from their similarity to known genes. We have first of all been concerned with solving this problem through clustering analysis. We have also discussed and demonstrated (in the "tumour and normal tissue" data set analysis) that modelling can improve this analysis. Clustering methods assign both known and unknown genes to clusters. Consequently the predictive aspects of a model alone is less important. However, if the predictive performance of a model is good, our belief in its descriptive quality tend to increase. A model can give us valuable descriptive knowledge about the typical features of a gene belonging to a certain cluster. This kind of information is not available from a clustering method. It simply gives us the clusters and nothing else. In the same way as we view syntactical and semantical clustering methods to be complementary, we also consider supervised and unsupervised learning methods to be complementary methods in the analysis of gene expression data.

When enough information is available about the known genes we can group them into clusters simply on the basis of this prior knowledge. In these cases we skip the clustering step in the KDD process and predict the function of unknown genes using a model induced from the classes of known genes. This method seems intuitively appealing since it avoids the difficulties of unsupervised learning in that it makes use of existing biological knowledge. As a steadily increasing number of genes becomes known this might be an even more powerful method in the future. However, one should be aware of the fact that one gene can be involved in more than one biological process. Hence we need knowledge not only about the gene itself, but also about this gene in the specific biological setting that we are studying.

9.1.5 Experiences from our Data Analysis

Our analysis of the Fibroblast data shows how syntactical and semantical methods can be used iteratively to improve the clustering results in co-operation with biological experts. It also shows the mutual dependencies between computer scientists and biologists in order to analyse data. Without computational methods and tools the data sets resulting from microarray experiments are simply too large to be analysed manually by the biologists. On the other hand, the computer-generated results are more or less worthless without a biological interpretation. Moreover, the co-operation with biologist throughout the analysis make us able to iteratively discuss and improve the results.

The biological interpretation of the analysis of the Fibroblast data is still going on. However, we have already been able to partly compare the results with existing knowledge. This, and the manual inspection of the clusters done by the biologist, have given us valuable information of the dynamics of the data set. A complete biological interpretation of the results will be presented elsewhere.

The analysis of the "tumour and normal tissue" data give us a glimpse of what can be done further with a set of validated clusters. The introduction of classifiers for predicting the function of unknown genes and for diagnosing diseases will become even more important as a steadily increasing number of genes becomes known with respect to biological function.

9.2 Conclusions

In Chapter 1 we discussed the possibilities of using both clustering methods and modelling methods to infer the function of unknown genes through their similarity (or relation) to known genes. In the thesis we have presented and validated a large variety of methods based on hierarchical strategies, iterative strategies and Boolean reasoning. Moreover, we have discussed how clustering and modelling can be used together in gene expression analysis. This was illustrated in Figure 1.1.

Our use of both clustering methods and modelling methods on real world gene expression data sets shows how powerful computational methods are in this analysis. In Chapter 1 we stated that the advantages of working with groups of similar genes rather than single genes would be decisive for the biologist's possibilities to obtain meaningful results from the analysis. Moreover, we stated that the access to existing knowledge would be a major time-saving factor. Our experiences with gene expression analysis in this thesis prove that these hypotheses indeed are correct. Our preliminary biological results show the validity of the clustering results. Also, they show that the amount of existing biological knowledge is insurmountable without the help of information systems to correlate this knowledge with the clustering results. We also experienced that the approach of representing knowledge in an ontology enable us to both automatically collect existing knowledge and to use this knowledge to validate our obtained clusters.

We conclude that the computational methods used in this thesis indeed are capable of making decisive contributions to the the analysis of gene expression data set.

9.3 Further work

Finally we present some of the problem still to be solved. Some of them are specific to this thesis while some of them are of more general nature.

A software system for cluster analysis: A system that supports some or all of the features depicted in Figure 1.1 in Chapter 1 should be implemented in a similar framework as the ROSETTA system. In fact, the supervised learning is already supported by the ROSETTA system. The new system would be a framework in which the different clustering algorithms would be supported by different preprocessing and validation tools. The algorithms used in this thesis are implemented as Perl programs with a common library.

The Fibroblast data: Some problems still remain to be solved in the Fibroblast data analysis. One of them is the appearance of spikes in the data that we think may be noise. Also, as already discussed, the biological interpretation of the results is not finished. Thus we can not rule out minor adjustments to the method.

Development of methods searching for optimal clustering parameters: One could, for example use a genetic algorithm (see e.g. [Vinterbo and Ohno-Machado, 1999]) to search for optimal definitions of the indiscernibility relation in terms of weighting the different attributes or in terms of strictness of the definition. Of course, this requires some measure that balances the number of clusters and the quality of each cluster. This problem is discussed throughout the thesis and still remains to be solved.

Bibliography

[Alon et al., 1999] Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., and Levine, A. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci. USA*, Vol. 96:6745–6750.

[Ashburner, 1999] Ashburner (1999). *The Ashburner Gene Ontology*. http://whitefly.lbl.gov/∼suzi/.

[Brown, 1990] Brown, F. M. (1990). *Boolean Reasoning: The Logic of Boolean Equations*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

[Brown et al., 1999] Brown, M. P. S., Grundy, W. N., Cristianini, N., Sugnet, C. W., Furey, T. S., Ares, M., and Haussler, D. (1999). Knowledge-based analysis of microarray gene expression data by using support vector machines. *PNAS*, No. 1, Vol. 97:262–267.

[Carr et al., 1997] Carr, D. B., Somogyi, R., and Michaels, G. (1997). Templates for looking at gene expression clustering. *Statistical Computing & Statistical Graphics Newsletter*, Vol. 97:20–29.

[Eisen et al., 1998] Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, Vol. 95:14863–14868.

[Fayyad et al., 1996] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*, Vol. 17, No. 3:37–54.

[Gallagher et al., 1995] Gallagher, M., Laver, M., and Mair, P. (1995). *Representative Government in Modern Europe.*

[Graphviz, 1999] Graphviz (1999). *Graphviz - graph drawing software*. http://www.research.att.com/sw/tools/graphviz/.

[Heidar and Berntsen, 1995] Heidar, K. and Berntsen, E. (1995). *Vesteuropeisk politikk. Partier, regjeringsmakt, styreform.* Oslo.

[Hvidsten et al., 1999a] Hvidsten, T., Bjanger, M. S., Komorowski, J., White, M. F., and Guanglai, B. (1999a). Fault diagnosis in rotating machinery using rough sets and rosetta. *EUFIT'99, European Congress on Intelligent Techniques & Soft Computing, Aachen, Germany*.

[Hvidsten et al., 2000] Hvidsten, T. R., Jenssen, T.-K., Komorowski, J., Lægreid, A., Sandvik, A., and Tjeldvoll, D. (2000). Template-based gene expression analysis. *To be presented as a poster at RECOMB2000, Computational Molecular Biology, Tokyo, Japan*.

[Hvidsten et al., 1999b] Hvidsten, T. R., Jenssen, T.-K., Lægreid, A., Øhrn, A., Tjeldvoll, D., and Komorowski, J. (1999b). Boolean reasoning in the analysis of gene expression data. *Presented as a poster at Data Mining for Bioinformatics - Towards In Silico Biology, Cambridge, UK.*

[Iyer et al., 1999] Iyer, V. R., Eisen, M. B., Ross, D. T., Schuler, G., Moore, T., Lee, J., Trent, J. M., Staudt, L., Jr., J. D., Boguski, M. S., Lashkari, D., Shalon, D., Botstein, D., and Brown, P. O. (1999). The transcriptional program in the response of human fibroblast to serum. *Science*, Vol. 283:83–87.

[Jenssen et al., 1999] Jenssen, T.-K., Lægreid, A., Komorowski, J., and Hovig, E. (1999). (re-) discovering gene-gene relations from textual data. *Presented as poster at Data Mining for Bioinformatics - Towards In Silico Biology, Cambridge, UK.*

[Jenssen et al., 2000] Jenssen, T.-K., Lægreid, A., Komorowski, J., and Hovig, E. (2000). Pubgen: Discovering and visualising gene-gene relations. *To be presented as a poster at RECOMB2000, Computational Molecular Biology, Tokyo, Japan*.

[Johnson, 1974] Johnson, D. S. (1974). Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, Vol. 9:256–278.

[Johnson and Wichern, 1998] Johnson, R. A. and Wichern, D. W. (1998). *Applied Multivariate Statistical Analysis*. Prentice-Hall, Inc.

[Kohonen, 1990] Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, Vol. 78, No. 9:1464–1480.

[Komorowski et al., 2000a] Komorowski, J., Pawlak, Z., Polkowski, L., and Skowron, A. (2000a). A rough set perspective on data and knowledge. *to appear in Handbook of Data Mining and Knowledge Discovery (W. Klösgen, J. Zytkow, Eds.), Oxford University Press*.

[Komorowski et al., 2000b] Komorowski, J., Skowron, A., and Øhrn, A. (2000b). Rosetta. *to appear in Handbook of Data Mining and Knowledge Discovery (W. Klösgen, J. Zytkow, Eds.), Oxford University Press*.

[Løken, 1999] Løken, T. (1999). *Rough Modeling: Extracting Compact Models from Large Databases*. Master thesis, Norwegian university of science and technology.

[Lowe et al., 1999] Lowe, A., Jones, R. W., and Harrison, M. J. (1999). Temporal pattern matching using fuzzy templates. *Journal of Intelligent Information Systems*, No. 13:27–45.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill Companies, Inc.

[Moxon, 1998] Moxon, B. (1998). Mining gene expression databases; visualization-based knowledge discovery for pharamaceutical and biotechnology companies. *Compaq Online Lab: www.eurpoe.digital.com/onlinelab/whitepapers*.

[Nguyen and Nguyen, 1996] Nguyen, H. S. and Nguyen, S. H. (1996). Some efficient algorithms for rough set methods. *Proc. Fifth Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'96)*, pages 1451–1456.

[Nguyen and Nguyen, 1998] Nguyen, H. S. and Nguyen, S. H. (1998). Discretization methods in data mining. *Rough Sets in Knowledge Discovery 1: Methodology and Applications, Physica-Verlag, Heidelberg, Germany*, pages 451–482.

[Nguyen and Skowron, 1995] Nguyen, H. S. and Skowron, A. (1995). Quantization of real-valued attributes. *Proc. Second International Joint Conference on Information Sciences*, pages 34–37.

[Nilsson, 1998] Nilsson, N. J. (1998). *Artificial intelligence: A New Synthesis*. Morgan Kaufmann Publishers, Inc., San Francisco, California.

[Pawlak, 1982] Pawlak, Z. (1982). Rough sets. *International Journal of Computer and Information Sciences*, Vol. 11:341–356.

[Pawlak, 1991] Pawlak, Z. (1991). *Rough Sets - Theoretical Aspects of Reasoning about Data*. Kluwer Academic Publishers, Dordrecht.

[Pellegrini et al., 1999] Pellegrini, M., Marcotte, E. M., Thompson, M. J., Eisenberg, D., and Yeates, T. O. (1999). Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles. *Proc. Natl. Acad. Sci. USA*, Vol. 96:4285–4288.

[Ripley, 1996] Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press.

[Russel and Norvig, 1995] Russel, S. and Norvig, P. (1995). *Artificial intelligence - a modern approach*. Prentice-Hall International, Inc.

[Schalkoff, 1992] Schalkoff, R. (1992). *Pattern Recognition - Statistical, structural and neural approaches*. John Wiley & Sons, Inc.

[Schena, 1999] Schena, M. (1999). *DNA Microarrays - A practical approach*. Oxford University Press.

[Sjøberg, 1998] Sjøberg, N. O. (1998). *Molekylær Genetikk. Genteknologi - humant DNA*. Vett og Viten.

[Skowron and Rauszer, 1992] Skowron, A. and Rauszer, C. (1992). The discernibility matrices and functions in information systems. *Intelligent Decision Support - Handbook of Applications and Advances of the Rough Sets Theory, Dordrecht, Kluwer Academic Publishers*, pages 331–362.

[Strachan and Read, 1999] Strachan, T. and Read, A. P. (1999). *Human Molecular Genetics*. BIOS Scientific Publishers Ltd.

[Struzik and Siebes, 1999] Struzik, Z. R. and Siebes, A. (1999). The haar wavelet transformation in the time series similarity paradigm. *Principles of Data Mining and Knowledge Discovery (PKDD'99), Prague, Czech Republic.*

[Tjeldvoll, 1999] Tjeldvoll, D. (1999). *Semi-automatic Gene Anotation*. Master thesis, Norwegian university of science and technology.

[Tjeldvoll et al., 1999] Tjeldvoll, D., Komorowski, J., White, M. F., and Guanglai, B. (1999). Analysis of ovarian tumor data. *EUFIT'99, European Congress on Intelligent Techniques & Soft Computing, Aachen, Germany*.

[Vinterbo and Ohno-Machado, 1999] Vinterbo, S. and Ohno-Machado, L. (1999). A genetic algorithm to select variables in logistic regression: example in the domain of myocardial infarction. *Journal of the American Medical Informatics Association*, pages 984–988.

[Vinterbo, 1999] Vinterbo, S. A. (1999). *Predictive Models in Medicine: Some Methods for Construction and Adaptation*. PhD thesis, Norwegian University of Science and Technology.

[Wen et al., 1998] Wen, X., Fuhrman, S., Michaels, G. S., Carr, D. B., Smith, S., Barker, J. L., and Somogyi, R. (1998). Large-scale temporal gene expression mapping of central nervous system development. *Proc. Natl. Acad. Sci. USA*, Vol. 95:334–339.

[Øhrn, 1999a] Øhrn, A. (1999a). *Discernibility and Rough Sets in Medicine: Tools and Applications*. PhD thesis, Norwegian University of Science and Technology.

[Øhrn, 1999b] Øhrn, A. (1999b). *ROSETTA, Technical Reference Manual*. The ROSETTA homepage: http://www.idi.ntnu.no/∼aleks/rosetta.

[Øhrn and Komorowski, 1999] Øhrn, A. and Komorowski, J. (1999). Diagnosing acute appendicitis with very simple classification rules. *Presented as a poster. Principles of Data Mining and Knowledge Discovery (PKDD'99), Prague, Czech Republic.*

[Øhrn et al., 1998] Øhrn, A., Komorowski, J., Skowron, A., and Synak, P. (1998). The design and implementation of a knowledge discovery toolkit based on tough sets: The rosetta system. *Rough Sets in Knowledge Discovery 1: Methodology and Applications, number 18 and Studies in Fuzziness and Soft Computing, Physica-Verlag, Heidelberg, Germany, chapter 19: 376–399*.

[Ågotnes, 1999] Ågotnes, T. (1999). *Filtering Large Propositional Rule Sets While Retaining Classifier Performance*. Master thesis, Norwegian university of science and technology.

[Ågotnes et al., 1999] Ågotnes, T., Komorowski, J., and Løken, T. (1999). Taming large rule models in rough set approaches. *Principles of Data Mining and Knowledge Discovery (PKDD'99), Prague, Czech Republic.*, pages 193–203.