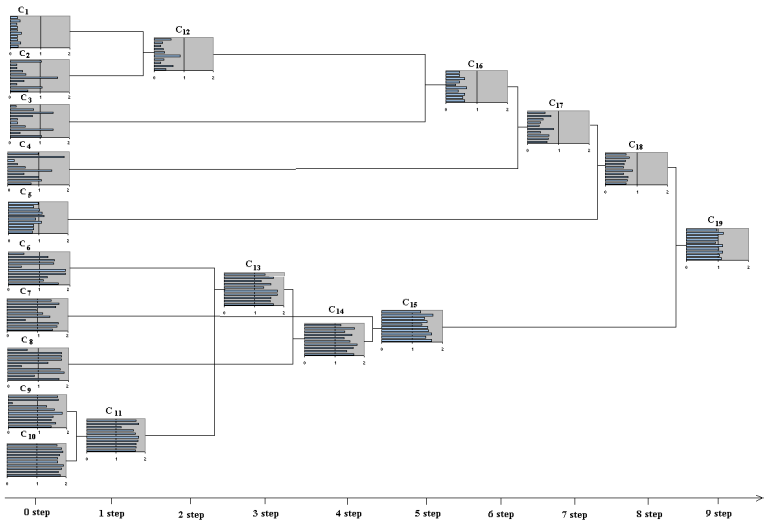
Model-based hierarchical clustering using Bayesian networks

Master Thesis



Student

Rasa Jurgelėnaitė

Supervisor

Jose M. Peña

Faculty of Engineering and Science
Aalborg University

Department of Computer Science

TITLE: Model-based hierarchical clustering using Bayesian networks
Master Thesis

SEMESTER PERIOD:
KDE4,
February 1st - June 10th, 2003

PROJECT GROUP:
E1-121d

GROUP MEMBERS:
Rasa Jurgelėnaitė, rasa@cs.auc.dk

SUPERVISOR:
Jose M. Peña, jmp@cs.auc.dk

NUMBER OF COPIES: 3

NUMBER OF PAGES: 51

SYNOPSIS:

This report introduces and empirically evaluates a model-based clustering algorithm, which uses directed graphical models (Bayesian networks) to describe the clusters and adopts hierarchical clustering strategy. Two model structure learning algorithms, the PC algorithm and the KES algorithm, are applied in the proposed clustering algorithm. The experimental results for gene expression data are reported evidencing the reasonable perfomance of the introduced approach.

Preface

This report has been written by project group E1-121d as documentation for the second part of a Master Thesis at the Department of Computer Science at Aalborg University. The report has been written during the period from the 1st of February to the 10th of June, 2003.

In the report we follow some general conventions which the reader should be familiar with before proceeding. The references in the report are marked with a number corresponding to the numbers in the literature list like [17]. Certain phrases are abbreviated. The abbreviation appears in parenthesis after the full phrase, for example "... Hierarchical Agglomerative Clustering (HAC) is ...". Attribute names in-lined in the text will be written in **bold font**. *Italics* are used to mark the introduction of a new technical term and the term can be expected to be explained short after.

I would like to thank my supervisor Jose M. Peña for his great help, guidance and valuable comments in this project.

Rasa Jurgelėnaitė

Contents

1 Introduction

This paper introduces a model-based clustering algorithm, which uses Bayesian networks as models, and documents investigation on its properties. As clustering is one of the thechniques used in data mining, this work can bee seen as a continuation of a data mining project implemented in cooperation with the textile company Green House during KDE3 course [18].

1.1 Data mining

Data mining or *knowledge discovery in databases* is a fairly young research area that has emerged as a reply to the flood of data we are faced with nowadays. It tries to meet the challenge to develop methods that can help human beings to discover useful patterns in their data. Data mining can be defined as the non-trivial extraction of implicit, previously unknown, and potentially useful information from data [11].

Data mining has its applications within science and research as well as in the industry and in business applications. It suits perfectly within application areas where there is a huge amount of factors each affecting the application area. The biology and medical research societies which deal with a huge amount of data such as DNA-profiles, diseases, symptoms, blood-types and drugs, have been using data mining with success. Data mining has also been dragging attention from the common business areas such as supermarkets and mobile phone providers. The following is an often referred example of a successful application of data mining performed by an American supermarket chain. It illustrates how the process of examining raw data, drawing mature conclusions and as a consequence, deploying the result in the business can result in an improved understanding of a business. Moreover, the resulting knowledge increased profit for the supermarket.

> "For example, one Midwest grocery chain used the data mining capacity of Oracle software to analyze local buying patterns. They discovered that when men bought diapers on Thursdays and Saturdays, they also tended to buy beer... The retailer concluded that they purchased the beer to have it available for the upcoming weekend. The grocery chain could use this newly discovered information in various ways to increase revenue. For example, they could move the beer display closer to the diaper display." [25]

1.2 Predictive and descriptive models

A successful data mining process applied on a database ends in knowledge, which is called a *model* and is a high level, global description of a data set. According to their final goal, data mining techniques can be considered to be *predictive* or *descriptive*.

A predictive model uses existing data to predict some response of interest. For instance, a credit card company may want to engage in a predictive model in order to identify the transactions that are thought most likely to be fraudulent. Thus, predictive models are learned by processing an existing set of data and should be able to predict some circumstances regarding yet unseen data.

As opposed to the predictive model, a descriptive model is not used to predict or classify yet unseen data. It tries to give a short summarized and human understandable description of an existing data set. Descriptive models are well suited to analyze characteristics of certain populations, e.g. in medical science or in business analyzing various factors concerning a group of people could make it possible to detect abnormalities, such as people who suffer from a certain disease or subpopulations requiring more specific marketing.

1.3 Clustering

One way to generate a descriptive model is by use of *clustering*. Clustering is the search for the description of group-structure underlying some given data. In some cases the description is a partition of the data, the other times is a model of the generative mechanism underlying the data.

Discovering groups in the data makes sense only if such groups exist. Therefore, clustering is based on the assumption that the data is generated by an underlying model which is responsible for such groups. Existing clustering approaches can be divided into two main categories: *partitional* approaches that attempt to find the groups of instances generated by the same model and *probabilistic/model-based* approaches that attempt to learn generative models from the data, with each corresponding to one particular cluster.

In general, two main problems are encountered while clustering. First, the number of underlying clusters has to be found and second, the clusters need a

description. We aim to propose a clustering algorithm that employes the model-based approach and can cope with the mentioned problems.

In this paper, we present a model-based hierarchical clustering algorithm that uses directed graphical models (Bayesian networks) to describe clusters. The method applies an agglomerative clustering procedure to discover the most probable set of clusters and employes a partitioning method to get the initial partitions. The introduced algorithm results in a dendogram where each cluster is represented by a Bayesian network, thus enables us to exploit the advantages of Bayesian networks.

The report is organized as follows. In Chapter 2, we give the necessary background in clustering. An overview of graphical models and particularly Bayesian networks as well as learning them from data is given in Chapter 3. In Chapter 4 we define the model-based hierarchical clustering algorithm using Bayesian networks. Chapter 5 discusses the experimental results for real-world data. We conclude in Chapter 6. The possible directions for future work are presented in Chapter 7.

2 Clustering

As described in Section 1.3 clustering is one of the possible ways to generate a descriptive model. In this chapter we will discuss clustering and its different approaches. k-means algorithm and hierarchical clustering techniques will be examined as they will be involved in the clustering algorithm proposed in Chapter 4.

2.1 Clustering

As we have already mentioned, clustering is the search for the description of group-structure underlying some given data where the description can be either a partition of the data or a model of the generative mechanism underlying the data. It is clear that it only makes sense to identify groups if some groups exists. Therefore, clustering is based on the assumption that the data is generated by an underlying model which is responsible for such groups. Specifically, the purpose of clustering is to gain more information about this model. Figure 2.1 depicts a mechanism which is often used to explain the underlying model. It consists of a *selector*, a number of physical processes and the data set. The assumption is that each instance in the data set is generated by this mechanism. For each instance the selector selects one and only one of the physical processes. The physical process then generates each attribute value of the instance, based on an unknown probability distribution. In the end, all the instances generated by one physical process are assumed to belong to the same cluster. The clusters and the physical processes remain unknown or hidden, i.e. it is unknown by which of the physical processes and how a specific instance was generated.

More specifically, cluster analysis in this report is based on the following assumptions:

1. Clustering is applied to a data set D containing N instances $d_1 \dots d_N$. Each instance d_i is a vector of p values $d_{i1} \dots d_{in}$ where each value belongs to one of the variables $A_1 \dots A_n$ of the data set.

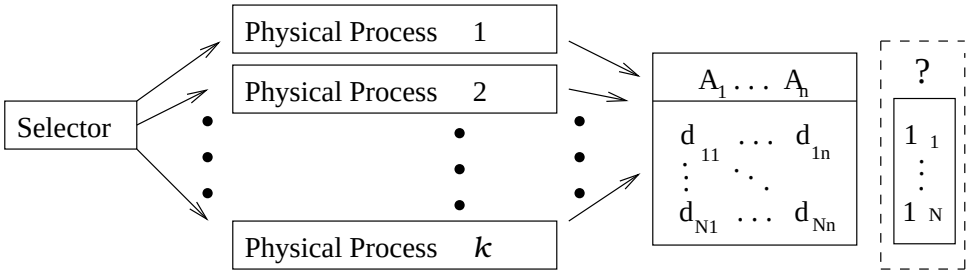


Figure 2.1: The underlying assumption: Each instance in the data set is generated by a physical process, selected by a selector which remains unknown for us. Physical processes as well as their number are also unknown.

2. Each instance $d_i \in D$ is a member of one and only one of the underlying hidden clusters $C = \{c_1, \ldots c_k\}$. The membership is represented by the label l_i assigned to each d_i. This label is unknown for us (hidden cluster membership).

3. D is generated by an underlying model consisting of k physical processes which, together with the selector are represented by a joint probability distribution.

In general, when clustering, one encounters two main problems. First the number of underlying clusters (k) has to be found and second, the clusters need a description. The description of the clusters, or the model, must be optimal with respect to some measurement function which assigns a score to each possible description. This measurement is based on an intuitive understanding of the term "being similar" which is referred to as the *clustering criterion*. If this criterion is translated into a mathematical formula which measures the homogeneity within each cluster, the clustering problem is left as a search for the description that yields the most homogeneous clusters. There exist two classical descriptions of the clusters, *partitional* and *probabilistic*, also called *model-based*. Partitional data clustering covers all the approaches which describe the data set by separating it into k non-empty, exhaustive and mutually exclusive subsets of instances which are similar to each other according to the clustering criterion. The probabilistic descriptions however, describe the clusters by modeling the mechanism that generated the data. After identifying a number of clusters it recovers the different probability distributions assigned to each of the physical processes and the selectors [27].

2.2 Partitional data clustering

Due to the lack of a priori knowledge of the mechanism that caused the instances grouped in D, partitional data clustering involves a very simple definition of the description of the clusters that exist in a data set D. Partitional data clustering describes each cluster of D by means of the set of instances. This way the partitional approach reduces data clustering to completing every instance of the original unlabelled database D with the label of the cluster whose physical process generated the instance.

Some researchers introduce a subdivision of the methods in partitional data clustering into groups of *partitioning* and *hierarchical* methods.

The partitioning methods attempt to directly decompose the data into disjoint clusters: first creates an initial partition of k groups, then it uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another.

In a hierarchical clustering the data are not partitioned into a particular number of clusters at a single step. Instead the clustering consists of nested partitions which may run from a single cluster containing all individuals, to N clusters each containing a single individual.

2.2.1 Partitioning algorithms

Further we will present k-means algorithm which is the most popular approach in data partitioning algorithms [14].

k-means algorithm

In k-means each cluster c_i is represented by a *cluster centroid*, $\overline{x_i}$, which is the n-dimensional mean vector of values $x_{i1}, \ldots x_{in}$ each corresponding to one of the attributes $A_1 \ldots A_n$. The clustering criterion which the k-means algorithm is based on is a distance measure. A common choice for continuous data is the Euclidean distance function. Other distance measurement functions may be more appropriate for different data types. In some cases the distance measurement function is called similarity measurement function refering to the fact that when the distance between a set of instances is low they are said to be similar.

The similarity measurement allows the assignment of an instance to a cluster with similar instances. This is determined by finding the nearest cluster centroid. The centroid can be defined as:

$$\overline{x_i} = (x_{i1}, \ldots x_{in}) = \frac{1}{N_i} \sum_{d \in c_i} d \tag{2.1}$$

where N_i is the number of instances in cluster i, $N_i > 0$, and d is a vector that represents an instance of cluster c_i. The formula calculates the centroid by summing the instances (vectors) of the clusters and dividing it by the number

of the instances in the cluster.

The pseudo-code for k-means algorithm follows:
The algorithm: k-means. For partitioning based on the centroids of the clusters.
Input: The number of clusters k and N instances of a database $(d_1, \ldots, d_N)$.
Output: A set of k clusters $(c_1, \ldots c_k)$.
Method:

```
select randomly k instances as initial cluster centroids;
repeat
  For every instance in the database and preserving the instances order Do
      Re(assign) the instance to the nearest cluster to it;
      If the instance changes its cluster, recalculate the
         centroids of the previous and current cluster;
until no instance can change its cluster or some stop criterion is met.
```

Drawbacks of k-means algorithm

Despite being widely used, the k-means algorithm is not exempt from drawbacks. The most important drawbacks, which are extensively reported in the literature are listed below:

- As many clustering algorithms, the k-means algorithm assumes that the number of clusters k in the database is known beforehand which, is not necessarily true in real-world applications.
- As an iterative technique, the k-means algorithm is especially sensitive to initial starting conditions: initial clusters and order of instances [21] [26].
- The k-means algorithm converges typically to a local minima. The running of the algorithm defines a deterministic mapping from the initial solution to the final one [34].

2.2.2 Hierarchical algorithms

There are two kinds of hierarchical clustering techniques: the agglomerative and the divisive. They construct their hierarchy in the oposite direction, possibly yielding quite different results. *Agglomerative* methods follow bottom-up strategy and start having all objects apart, that is in the beginning we have N clusters. Then in each step two cluster are merged, until only one is left. On the other hand, *divisive* methods follow top-down approach when in the beginning all objects are in one cluster and each following step a cluster is split up, until there are N of them. Hierarchical clusterings may be represented by a two-dimensional diagram known as *dendrogram* which illustrates the fusions or divisions made at each successive stage of analysis. An example of such a diagram for data set with $N = 5$ instances is given in Figure 2.2.

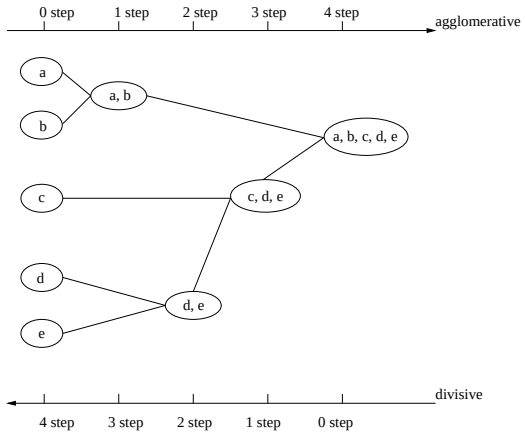


Figure 2.2: Hierarchical clustering of the set $\{a, b, c, d, e\}$

Of the two types of hierarchical clustering scheme, the agglomerative methods are far more widely used. Divisive algorithms have been much less used than agglomeratives due to, in principle, their much bigger computational complexity and requirements for computer resources. For example, the first step of the agglomerative algorithm is to check which pair of objects should be merged and makes a total of:

$$\frac{N(N-1)}{2} \tag{2.2}$$

evaluations. However, the first step of divisive algorithm has to look for all possible splittings into two and makes a total of:

$$2^{N-1} - 1 \tag{2.3}$$

evaluations. Obviously it is impossible to make it with databases of relatively large size. Because of this reason divisive algorithms have received less studies and interest from the researchers [20]. From now on we focus only on agglomerative algorithms.

An agglomerative hierarchical clustering procedure produces a series of partitions of the data, $P_N, P_{N-1}, ..., P_1$. The first, P_N, consists of N single-member clusters while the last, P_1, consists of a single group containing all N individuals. At each particular stage the methods fuse individuals or groups of individuals which are closest (or most similar).

The basic operation of hierarchical agglomerative clustering (HAC) procedures is outlined:

1. Start by assigning each instance to its own cluster, compute distances (similarities) between the formed clusters.

2. Find the closest (most similar) pair of clusters and merge them into a single cluster and decrement the number of clusters by one.

3. Compute distances (similarities) between the new cluster and each of the old clusters.

4. Repeat steps 2 and 3 until all instances are clustered into a single cluster.

Differences between methods arise because of the different ways of defining distance (or similarity) between an individual and a group containing several individuals, or between two groups of individuals.

Single linkage clustering

One of the simplest agglomerative hierarchical clustering method is *single linkage* also often known as the *nearest neighbour* technique [9]. The defining feature of the method is that distance between groups is defined as that of the closest pair of individuals, where only pairs consisting of one individual from each group are considered. This measure of inter-group distance is illustrated in Figure 2.3. The single linkage method is closely related to certain aspects of graph theory.

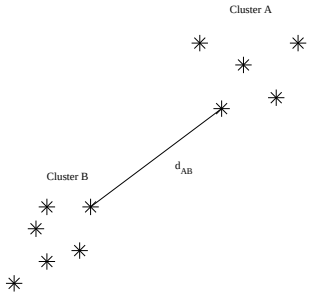


Figure 2.3: Single linkage distance

Complete linkage clustering

The *complete linkage* or *furthest neighbour* clustering method is the opposite of single linkage in the sense that distance between groups is now defined as that of the most distant pair of individuals, one from each group. The measure is illustrated in Figure 2.4.

Group-average clustering

Here the distance between two clusters is defined as the average of the distances between all pairs of individuals that are made up of one individual from each group.

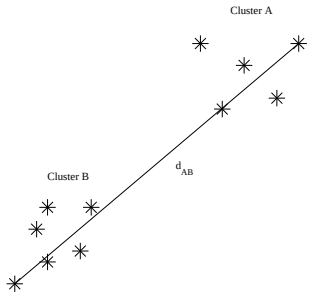


Figure 2.4: Complete linkage distance

Centroid clustering

With this method, groups once formed are represented by their mean values for each variable, that is, their *mean vector*, and inter-group distance is now defined in terms of distance between two such mean vectors.

A disadvantage of this method is that if the sizes of the two groups to be fused are very different then the centroid of the new group will be very close to that of the larger group and will remain within that group. The characteristic properties of the smaller group then are virtually lost.

Median clustering

The method assumes that the groups to be fused are of equal size. The apparent position of the new group will then always be between the two groups to be fused. Moreover if the centroids of the groups to be fused are represented by c_i and c_j, then the distance of the centroid of the third group c_h from the group formed by the fusion of c_i and c_j lies along the median of the triangle defined by c_i, c_j and c_h.

Ward's hierarchical clustering method

Ward [39] proposed a clustering procedure seeking to form the partitions P_N, $P_{N-1}, \ldots, P_1$ in a manner that minimizes the loss associated with each grouping, and to quantify that loss in a form that is readily interpretable. At each step in the analysis, the union of every possible pair of clusters is considered and the two clusters whose fusion results in the minimum increase in information loss are combined.

Information loss is defined by Ward in terms of an error sum-of-squares criterion, ESS. The rationale behind Ward's proposal can be illustrated most simply by considering univariate data. The loss of information having univariate data is given by

$$ESS = \sum_{i=1}^{k} \sum_{x_j \in c_i} (x_j - \overline{x_i})^2, \tag{2.4}$$

where k is the number of clusters, c_i is cluster i, $\overline{x_i}$ denotes a mean of cluster i.

Properties and problems of agglomerative hierarchical clustering techniques

A hierarchical method suffers from the defect that it can never repair what was done in previous steps. Once an agglomerative algorithm has joined two objects, they cannot be separated anymore. This rigidity of hierarchical agglomerative methods is both the key to its success as it leads to small computation times and its main disadvantage - the inability to correct erroneous decisions. On the other hand, hierarchical techniques do not really compete with partitioning methods because they do not pursue the same goal, as they try to describe the data in a totally different way [19].

Empirical studies point to Ward's method, group average and complete linkage as the most useful in practice, although the results are not clear out, and it was found that no single method could be claimed superior for all types of data [9].

It is often the case, when hierarchical clustering techniques are used in practice, that the investigator is not interested in the complete hierarchy but only in one or two partitions obtained from it. In hierarchical clustering, partitions are achieved by 'cutting' the dendrogram or selecting one of the solutions in the nested sequence of clusterings that comprise the hierarchy. A number of applications have been suggested for this purpose but it remains a difficult problem.

2.3 Model-based data clustering

Despite the partitional approach to data clustering is broadly considered as a useful tool in many fields, it is well known that it sometimes suffers from the lack of theoretical basis. Thus model-based approach to data clustering gains an advantage over partitional approach due to its statistically well established root [1].

In model-based data clustering, the objective is to learn generative models from the data with each model component corresponding to one particular cluster. Model-based data clustering does not perform a completion of the unlabelled database D as every case may be seen as belonging to each of the k clusters with certain probability, due to the probabilistic nature of their description.

When facing data clustering from the model-based perspective, the data clustering criterion is usually a measure of closeness between the true joint probability

distributions that represent the mechanism that produced the data and the set of joint probability distributions being evaluated as description of this mechanism. As the true joint probability distributions are not available, the closeness is indirectly measured with the help of D. That is, the data clustering criterion usually reduces to assesing how likely it is that D was generated by the set of joint probability distributions being evaluated as description of the underlying clusters of D [28].

The most classical solution to model-based data clustering is based on the theory of *finite mixture models* [8]. Despite the solid statistical background of finite mixture models in general and for probabilistic data clustering in particular, the high-dimensional manipulations and computations that they usually involve make them impractical in some cases and unattractive in some others. One of the paradigms that can help us to prevent the referred drawbacks and encode a joint probability distribution is the Bayesian network paradigm [17].

It is often the case that model-based clustering algorithms incorporate partitional clustering methods. As we mentioned in Chapter 1 the model-based clustering algorithm we propose will employ HAC approach for finding the most probable set of clusters. Involvement of HAC in model-based data clustering is not a novel idea. Model-based HAC algorithm has been explored by Banfield and Raftery [2] and Fraley [10] using Gaussian model components, by Vaithyanathan and Dom [38] using multinomial models for clustering documents. Ramoni et al. [31] present a Bayesian method for clustering dynamic processes where they apply a HAC to discover the most probable set of clusters capturing different dynamics.

The forthcoming chapter introduces Bayesian networks.

3 Bayesian networks

It is important to have knowledge discovering techniques that allow flexibility in the way knowledge can be encoded, represented and discovered. Probabilistic graphical models offer such a technique as they are a framework for structuring, representing and decomposing a problem using the notion of conditional independence.

We start this chapter by a short introduction to graphical models. Later on we only discuss directed graphical models, i.e. Bayesian networks.

3.1 Graphical models

Probabilistic graphical models are graphs in which nodes represent random variables, and the (lack of) arcs represent conditional independence assumptions. Hence they provide a compact representation of joint probability distributions. Graphical models are subdivided into *undirected* and *directed* according to their definition of independence. In undirected graphical models, also called Markov random fields or Markov networks, conditional independence is based on presence or absence of arcs. The notion of independence for directed graphical models, also called Bayesian networks or belief networks, is more complicated as it takes into account the directionality of arcs.

Although directed graphical models have a more complicated notion of independence than undirected models, they have several advantages. The most important one is that one can regard an arc from A to B as indicating that A causes B. This can be used as a guide to construct the graph structure when the structure is learned by domain experts. In addition, directed models can encode deterministic relationships and are easier to learn (fit to data) [23].

Graphical models are popular because of graphs that provide a structural view

of a probability distribution without getting lost in the mathematical details, thus make graphical models easily understood by human beings. Another nice property that distinguishes graphical models from other approaches (such as regression, decision trees, rules) is that they are independent of the choice of the target variable, so they can be used as a problem independent knowledge representation.

3.2 Bayesian networks

A *Bayesian network* (BN) for a random variable $X = (X_1, ..., X_n)$ consists of:

- A *directed acyclic graph* (DAG) G over X encoding the conditional (in)dependencies between the random variables of X (i.e., model structure).
- A set of parameters Θ_G for the local probability distributions imposed by the model structure (i.e., model parameters).

The semantics of a BN is that the joint probability distribution of X can be factored into the product of conditional probability distributions of each variable X_i given the set of its parents π_i. A BN represents a joint probability distribution for X as follows:

$$p(x|\Theta_G, G) = \prod_{i=1}^{n} p(x_i|x_{\pi_i}, \Theta_i, G). \tag{3.1}$$

This factorization formalizes the graphical intuition that X_i depends on its parents: given its parents, X_i is conditionally independent of all other variables which are not descendants of X_i. The set of parameters governing the conditional distribution which relates X_{π_i} to X_i is denoted by Θ_i, while the set of all parameters in the BN is denoted $\Theta_G = (\Theta_1, ..., \Theta_n)$.

Consider the example in Figure 3.1. Here, nodes represent random binary variables. We can see that the event "grass is wet" $(W = true)$ has two possible causes: either the water sprinkler is on $(S = true)$ or it is raining $(R = true)$. The strength of this relationship is shown in the model parameters. For example, we see that $P(W = true|S = true, R = false) = 0.9$ (second entry of second row), and hence, $P(W = false|S = true, R = false) = 1 - 0.9 = 0.1$, since each row must sum to one. Since the node C has no parents, the prior probability for being cloudy is specified (in this case, 0.5).

The simplest conditional independence relationship encoded in a BN can be stated as follows: a node is independent of its ancestors given its parents, where the ancestor/parent relationship is with respect to some fixed topological ordering of the nodes.

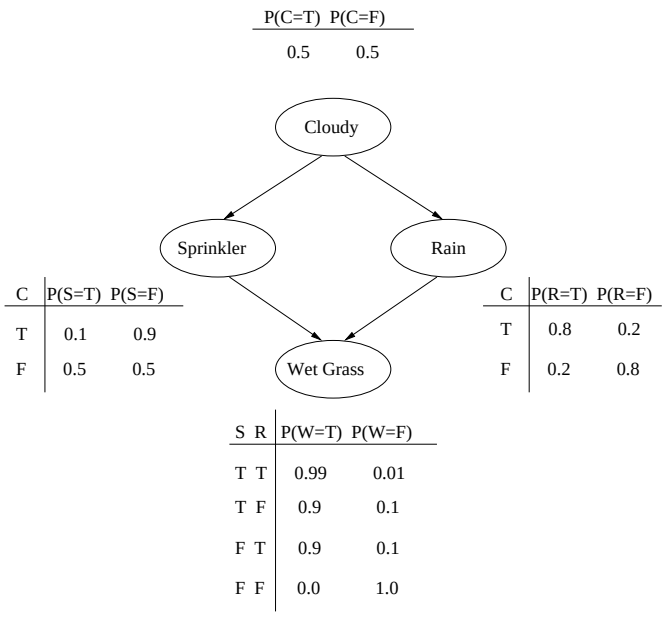


Figure 3.1: A simple BN

By the chain rule of probability, the joint probability of all the nodes in Figure 3.1 is

$$P(C, S, R, W) = P(C) * P(S|C) * P(R|C, S) * P(W|C, S, R) \tag{3.2}$$

By using conditional independence relationships, we can rewrite this as

$$P(C, S, R, W) = P(C) * P(S|C) * P(R|C) * P(W|S, R) \tag{3.3}$$

where we were allowed to simplify the third term because R is independent of S given its parent C, and the last term because W is independent of C given its parents S and R.

BNs have two main advantages over other paradigms for knowledge/uncertainty representation. These are provision of insight into the conditional (in)dependencies in the problem domain and ability to perform efficient and effective reasoning under uncertainty.

The ability to combine two different sources of information, domain experts and data, for learning structure and parameters is another quality of BNs.

3.3 Learning Bayesian networks from data

Learning a BN can be separated into two tasks: (1) *structure learning*, i.e., identifying the topology of the network, and (2) *parameter learning*, i.e., determining the associated joint probability distribution for a given network topology [30]. The former task is considered to be more challenging than the latter.

Two main sources of information to learn structure and parameters of a BN exist: domain experts and data. Most often, however, the structure is provided by domain experts alone. The probability tables, on the other hand, are often generated from data using a variety of statistical methods. However, this does not mean that the probabilities may only be based on statistics. Equally, the probability tables can be generated on the base of subjective assessments by domain experts. Combining expert knowledge and statistical methods often provides the best practical solution to constructing robust models.

In the remainder of this report, we will only discuss learning BNs from data. Moreover, we restrict our discussion to only learning BNs from complete data as we assume that data have no missing values.

3.3.1 Structure learning

The goal of structure learning is to learn a DAG that best explains the data. It was shown that the problem is NP-hard since the number of DAGs on n variables is super-exponential in n [4].

The model structure learning algorithms can be divided into two main groups: *dependency analysis* algorithms based on detecting conditional (in)dependencies and *search and scoring/model selection* algorithms based on problem optimization, according to their nature.

Dependency analysis

The dependency analysis approach performs a series of tests of conditional independence on the sample, and uses the results to construct the set of DAGs that most closely implies the results of tests. If the time order of the variables is known, the output is a single DAG.

Probably the most spread BN structure learning algorithm based on detecting conditional (in)dependencies is *PC algorithm* introduced by Spirtes and Glymour [37].

PC algorithm

The PC algorithm learns a BN by starting from a complete undirected graph. In the first pass, the algorithm removes each link if the end nodes of the link are marginally independent. In the second pass, it removes each link if the end nodes of the link are independent conditioned on a third node. In each of the following passes, it removes each link if the end points of the link are independent conditioned on a subset of nodes of higher order until a stopping condition is met. Finally, the arcs are oriented in order to obtain a DAG.

Let Adjacencies(C, A) be the set of vertices adjacent to A in the undirected acyclic graph C. In the algorithm, the graph C is continually updated, so

Adjacencies(C, A) is constantly changing as the algorithm progresses. Pseudo-code for PC algorithm follows:

```
(A) Form the complete undirected graph C on vertex set V.

(B) n=0.
    repeat
       repeat
          select an ordered pair of variables X and Y that are adjacent in C
          such that Adjacencies(C,X)\{Y} has cardinality greater than or equal
          to n, and a subset S of Adjacencies(C,X)\{Y} of cardinality n, and
          if X and Y are conditionally independendent given S delete edge
          X - Y from C and record S in Sepset(X,Y);
       until all ordered pairs of adjacent variables X and Y such that
       Adjacencies(C,X)\{Y} has cardinality greater than or equal to n and all
       subsets S of Adjacencies(C,X)\{Y} of cardinality n have been tested for
       conditional independence;
       n=n+1;
    until for each ordered pair of adjacent vertices X, Y, Adjacencies(C,X)\{Y}
    is of cardinality less than n.

(C) For each triple of vertices X, Y, Z such that the pairs X, Y and Y, Z are
    each adjacent in C but the pair X, Z is not adjacent in C, orient
    X - Y - Z as X -> Y <- Z if and only if Y is not in Sepset(X,Z).

(D) repeat
       If A -> B, B and C are adjacent, A and C are not adjacent, and there is
       no arrowhead at B, then orient B - C as B -> C.
       If there is a directed path from A to B, and an edge between A and B,
       then orient A - B as A ->B.
    until no more edges can be oriented.
```

Even if this group of algorithms is closer to the semantics of BN structures, the vast majority of BN structure learning algorithms belong to the model selection approach. These methods are presented further.

Model selection

The space search approach, which is also called model selection, searches the model that maximizes a certain scoring function in a space of models.

Model selection procedure usually consists of three components: (1) a *neighborhood*, (2) a *scoring criterion* and (3) a *search strategy*. The neighborhood of a model restricts the search to a small part of the search space around that model. The scoring cryterion evaluates the quality of a model and it is usually required to be score equivalent, locally consistent and decomposable. The search strategy selects a new model, from those in neihgborhood of the current best model, based on the scoring cryterion.

Different options for the components of model selection procedure are reviewed further.

Neighborhood

The neighborhood of a model defines the search space where the search is done. One of possible search spaces is the *inclusion boundary* $IB(M(G))$ which was characterized by Chickering [5] as the set of models represented by all those DAGs that can be obtained by adding or removing a single arc from any DAG G^* equivalent to G. Two DAGs are equivalent G_1 and G_2 are *equivalent* if they represent the same model, i.e. $M(G_1) = M(G_2)$.

Scoring criteria

From the Bayesian perspective, a natural scoring function is the marginal likelihood of model given data. Cooper and Herskovits gave a formula for computing it in the case of complete data [6]. Herewith they showed that exact computation of the score is intractable when missing data are present. In such cases asymptotic approximations of the marginal likelihood such as the *Bayesian Information Criterion* (BIC) [36] and the *Cheeseman-Stutz Criterion* (CS) [3] are usually employed. Hereinafter we will only discuss BIC which is widely used in BN structure learning.

The BIC score has two parts: one evaluates the fit of the model to the data and the other penalizes the model according to its complexity. The complexity of a model is measured by the number of independent parameters. The formula for computing the BIC score is presented below:

$$\sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\log\frac{N_{ijk}}{N_{ij}} - pe(N)dim(S) \tag{3.4}$$

where

- N_{ijk} denotes the number of cases in D in which X_i has the value x_i^k and π_i is instantiated as its j-th value,
- r_i denotes the number of states X_i has and q_i denotes the number of its parental combinations,
- $$N_{ij} = \sum_{k=1}^{r_i} N_{ijk},$$
- $$dim(S) = \sum_{i=1}^{n} q_i(r_i - 1),$$

- $$pe(N) = \frac{1}{2} \log N.$$

As we noted before, usually it is a requirement for a scoring criterion to be score equivalent, locally consistent and decomposable.

Suppose a BN model. Data can be used to select among different models according to some scoring criterion that assigns a score $S(M)$ to a model M. In some cases it can be convenient to assign a score $S(G) = S(M(G))$ to a representative DAG of a model, too. We say that the scoring criterion is *score equivalent* if it assigns the same value to all the DAGs representing the same model. A scoring criterion is *locally consistent* if the score given to a DAG for some data sampled from a joint probability distribution p assimptotically always increases by removing an arc in G, unless this arc removal adds a conditional independence constraint to the model that does not hold in p. Finally, we call a scoring criterion decomposable if it can be expressed as a sum over all the families of BN, where a family is a node and its parents.

BIC score is equivalent, locally consistent and decomposable.

Search algorithms

A variety of methods of search for models with the highest score have been proposed, including K-2, local search, hill-climbing, genetic algorithms, and simulated annealing.

As it was noted earlier structure learning is an NP-hard problem. However, the task is much simpler if the ordering of nodes is known, since we can learn the parent set for each node independently (since score is decomposable), and there is no problem of acyclity anymore. The algorithm that uses the assumption of total ordering among variables is K2 algorithm, developed by Cooper and Herskovits [6]. This algorithm is frequently used for BN structure construction from data.

Given a database D, K2 algorithm searches for the BN structure G that maximises the probability $P(G|D)$. K2 is a greedy heuristic. The algorithm starts by assuming that a node lacks parents, after which in every step it adds incrementally that parent whose addition most increases the probability of the resulting structure. K2 stops adding parents to the nodes when the addition of a single parent cannot increase the probability.

Despite being widely used the requirement of K2 algorithm for a total order among nodes to start, which can be regarded as a form of prior domain knowledge, is its significant drawback.

In this type of algorithms asymptotically optimal learning algorithm, greedy equivalence search algorithm [5], has been developed under the faithfulness assumption. As an alternative for this algorithm the k-greedy equivalence search algorithm (KES) [24] has been proposed.

The main characteristic of KES is that it allows a trade-off between greediness and randomness, thus exploring different good local optima when run repeatedly. Due to the possibility of trading between greediness and randomness KES can be seen as a family of algorithms. The KES algorithm uses inclusion boundary neighborhood, score equivalent, locally consistent and decomposable scoring criterion and complete data.

KES is formally described as follows:

```
M = empty graph model
repeat
   B = set of models in IB(M) with higher score than the model M
   if |B| > 0 then
      C = random subset of the set B with size max(1,|B|*k)
      M = the highest scoring model from the set C
   else return(M)
```

where k $\in$ [0,1].

The scoring and dependency analysis approaches have also been combined in the hope of obtaining a method that is more powerful than its constituents. For example, the CB algorithm does not require a total order among nodes as start; it uses conditional independence tests for that purpose, and uses K2 in its second phase [35].

3.3.2 Parameter learning

As it was already mentioned learning parameters of a BN is an easier task than learning structure of a BN.

In the case we have to learn model parameters from complete data, we assume that the goal of learning is to find the maximum likelihood (ML) estimates of the parameters of each conditional probability table, i.e., the parameter values which maximize the likelihood of the data.

Assume we are given a data set D of N independant and identically distributed observations of the settings of all variables in our BN structure $D = x^{(1)}, ..., x^{(N)}$, where $x^{(i)} = (x_1^{(i)}, ..., x_n^{(i)})$. The *likelihood* is a function of the parameters which is the probability of the observed data:

$$p(D|\Theta) = \prod_{i=1}^{N} p(x^{(i)}|\Theta) \tag{3.5}$$

We assume that the parameters are unknown and we wish to estimate them from data. We focus on the problem of estimating a single setting of the parameters which maximizes the likelihood. Equivalently we can maximize the log likelihood:

$$L(\Theta) = \log p(D|\Theta) = \sum_{i=1}^{N} \log p(x^{(i)}|\Theta) = \sum_{i=1}^{N} \sum_{j=1}^{n} \log p(x_j^{(i)}|x_{\pi_j}^{(i)}, \Theta_j) \quad (3.6)$$

where the last equality makes use of the factorization of joint distribution in the directed graphical model. If we assume that the parameters Θ_j governing the conditional probability distribution of X_j given its parents are functionally independent of the parameters governing the conditional probability distribution of other nodes in the graphical model, then the log likelihood decouples into a sum of local terms involving each node and its parents:

$$L(\Theta) = \sum_{j=1}^{n} L_j(\Theta_j) \quad (3.7)$$

where $L_j(\Theta_j) = \sum_{i=1}^{N} \log p(x_j^{(i)}|x_{\pi_j}^{(i)}, \Theta_j)$. Each L_j can be maximized independently as a function of Θ_j. For example, if the X variables are discrete and Θ_j is the conditional probability table for x_j given its parents, then the ML estimate of Θ_j is simply a normalised table containing counts of each setting of X_j given each setting of its parents in the data set.

If there is a small number of cases compared to the number of parameters prior can be used to regularize the problem. In this case, we call the estimates maximum a posteriori (MAP) estimates.

Maximum a posteriori parameter estimation incorporates prior knowledge about the parameters in the form of a distribution $p(\Theta)$. The goal of MAP estimation is to find the parameter setting that maximizes the posterior over parameters, $p(\Theta|D)$, which is proportional to the prior times the likelihood. If the prior factorizes over the parameters goverining each conditional probability distribution, i.e., $p(\Theta) = \prod_{j=1}^{n} p(\Theta_j)$ then MAP estimates can be found by maximizing

$$L'(\Theta) = \sum_{j=1}^{n} (L_j(\Theta_j) + \log p(\Theta_j)). \quad (3.8)$$

The log prior can be seen as a regularizer, which can help reduce overfitting in situations where there is insufficient data for the parameters to be well-determined.

4 Proposed algorithm

This chapter introduces a model-based clustering algorithm that uses BNs and employes the hierarchical agglomerative clustering approach. Different options for the particular steps of the algorithm will be discussed.

4.1 The algorithm

We introduce the model-based clustering algorithm that applies a hierarchical agglomerative procedure to describe the underlying cluster-structure of a given database. All clusters are represented by BNs which are well-known for their ability to structure, represent and decompose a problem using the notion of conditional independence.

The proposed algorithm performs a search by recursively merging the closest BNs representing a cluster and evaluating whether the resulting global model is more probable than the global model where these BNs represented two different clusters. A similarity measure between two BNs and a stopping criterion are used to guide this process.

The pseudo-code for the proposed model-based clustering algorithm using BNs is presented:

The algorithm: Model-based clustering algorithm using BNs.
Input: A database of N instances $d_1, \ldots, d_N$ and the number of initial clusters k.
Output: Nested partitions where each cluster is described by a BN.
Method:

```
1. Get k initial clusters.
2. Learn a BN for each cluster.
3. Compute the similarity between any pair of models.
4. Repeat
```

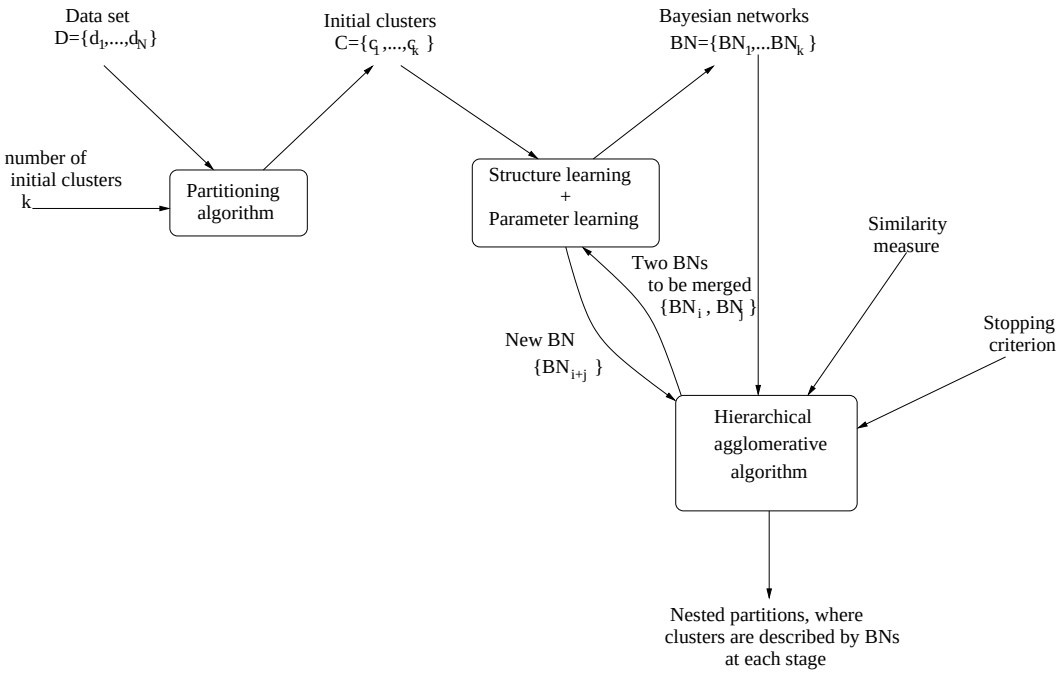


Figure 4.1: Graphical representation for the proposed algorithm

```
a. Merge the two closest clusters.
    b. Learn a new model using the instances from the newly merged cluster.
    c. Re-compute the similarity between the newly learned model and the
       other intact models.
  until stopping criterion is met.
```

Graphical representation for the proposed algorithm is given in Figure 4.1.

The model-based hierarchical clustering algorithm results in a dendogram where each cluster is represented by directed graphical model. The created models allow the experts of field to read conditional independencies from them. One more advantage of the proposed algorithm is evidence propagation, sum propagation and max propagation, what cannot be done if you don't take model-based clustering approach.

The algorithm we propose is similar to a Bayesian method for clustering dynamic processes introduced by Ramoni and Sebastiani [31]. The method models dynamics as first-order Markov chains and then applies a hierarchical agglomerative clustering algorithm to discover the most probable set of clusters capturing different dynamics. The significant difference between these two clustering algorithms is data: we aim to learn generative models from static, i.e. non-temporal, data.

In this paper we are only concerned with discrete data as we deal with discrete data clustering.

4.2 Initial clusters

The first step of the algorithm is getting initial clusters that could be later described by BNs. Partitioning methods suit perfectly for this task as they attempt to directly decompose data into disjoint clusters. k-means algorithm, which is often given as the most popular approach in the group of partitioning algorithms, can be used for getting initial clusters. k-means algorithm was introduced in Chapter 2.

As it was noted in Chapter 2, hierarchical agglomerative clustering provides a representation of the various clusters as branches of a tree, allowing very easy interpretation. The clear structure of the result visualization is one of the biggest advantages of this approach. However, especially with large datasets it is easy to lose the overview of the resulting tree representation [32]. This fact and a request for BN not to be overfitted because of too small quantity of data should be the main guidelines choosing the number of initial clusters.

4.3 Learning of Bayesian networks

For learning of BNs from data any of existing algorithms could be used. We decided to apply two algorithms representing two main groups of structure learning algorithms: PC algorithm, which represents dependency analysis algorithms and KES algorithm, which represents model selection algorithms. Both algorithms were introduced in Chapter 3. Since KES allows a trade-off between greediness and randomness we can be flexible setting its parameter.

As we do not intend to incorporate prior knowledge, learning parameters of BNs is simply the task of finding the maximum likelihood estimates of the parameters of each conditional probability distribution. Learning parameters of a BN was introduced in Chapter 3.

4.4 Similarity measure

The similarity measure between two BNs representing clusters could be any distance between probability distributions. Measures of distance such as symmetric Kullback-Leibler distance could be used. However, Kullback-Leibler approach may be inefficient for domains with a large number of attributes as it requires the joint probability of any state.

Instead we choose another approach to compute the similarity measure that would guide the merging process; we compute the fitness of model given the data. The similarity measure we propose is a computation of how well one model fits the data of the other model we tend to merge and vice versa. We look for the pair of models with the smallest value of similarity measure:

$$min_{i \neq j} = \frac{\frac{\log L(D_i|BN_j)}{N_i} + \frac{\log L(D_j|BN_i)}{N_j}}{2} \tag{4.1}$$

In the formula above the average fitness of the model given the data for one instance is computed.

If we are dealing with real-world data it is very likely that data are noisy. Thus, similarity measure can be untruthful as noisy instances are involved in computation. Using only those instances that are the best representatives of the model they belong to, could be a solution to avoid inaccurate results.

A possible way to select the best representatives of the model is to compute the probabilities for each instance entering the evidence to the model this instance belongs to. Later instances are ranked according to their probabilities and the chosen percentage of instances with the highest probabilities are used for computation of the similarity measure. In the terms of hierarchical agglomerative clustering approach could be a combination of group-average clustering and centroid clustering methods.

4.5 Stopping criterion

The task of the stopping criterion is to evaluate whether the resulting partition with two BNs merged into one is more probable than the previous partition where these BNs are kept apart.

The stopping criterion we introduce is based on the BN paradigm. A partition of clusters is represented by a *global model* that can be seen as a BN of one node where all BNs representing clusters are the states of this node. The proposed stopping criterion learns how well the global model fits data.

The scoring metric for evaluation and comparison of global models can be the BIC score for the global model:

$$BIC_{global} = \sum_{i=1}^{n_{st}} BIC_i + \sum_{i=1}^{n_{st}} N_i \log \frac{N_i}{N} - \frac{1}{2} \log N(n_{st} - 1) \tag{4.2}$$

where n_{st} denotes a number of states in the global model, i.e. the number of clusters left at this particular stage of clustering.

BIC_{global} consists of two components: the summation of BIC score of each state of the global model, i.e. each BN included in the global model, and estimate of the score for the global model itself.

Graphical representation of global models comparison is given in Figure 4.2.

Suppose we are at step 2 of the clustering process shown in Figure 4.2. We want to assess whether or not the model constructed from the current model such c_2 and c_3 are merged is better. If BIC_{global} score for a new global model is

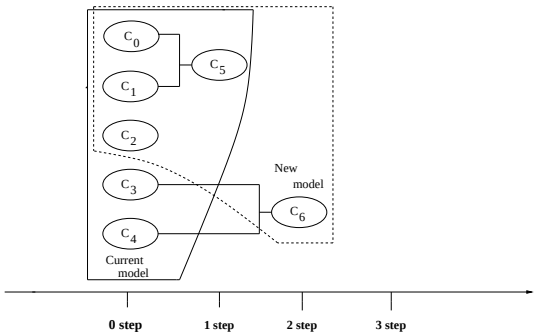


Figure 4.2: Comparison of global models

higher than BIC_{global} for a current model stopping criteria advices to continue on merging, otherwise - current model is recomended as the best solution.

The twofold task is set for the introduced stopping criterion: annotate the process of fusions and advise the most probable partition of clusters.

Experimental evaluations of the model-based clustering algorithm using BNs will be shown in the next chapter.

5 Experimental evaluation

In this chapter we empirically evaluate the effectiveness of the proposed clustering algorithm. Experiments are run with real-world data, gene expression data. We start the chapter by briefly describing the implementation of the algorithm. Introduction of data involved in the evaluation follows. Finally, we report and discuss the obtained results.

5.1 Implementation

In order to evaluate the proposed algorithm in practice certain settings have to be adopted. We report only the major settings involved in the implementation.

The performance of the algorithm highly depends on the set of initial clusters. As we reported already we chose k-means algorithm to discover the initial clusters. It is widely reported that k-means algorithm suffers from initial starting condition effects. Thus, we have adopted RANDOM initialization method, which divides the database into a partition of k clusters at random, as this method was shown to be a good election because of its good performance [26].

Once the set of initial clusters is found the next step is to describe each cluster by a BN. As it was noted in the previous chapter PC algorithm and KES algorithm were chosen to learn structure of BNs.

For running experiments with the PC algorithm (see Section 3.4 for the description) the implementation of the algorithm in the BN package Hugin was used [15]. The PC algorithm performs the dependency tests that calculate a test statistic assuming (conditional) independence. If the test statistic is large, the independence hypothesis is rejected; otherwise, accepted. While running the experiments we set the significance level, i.e. the probability of rejecting a true independence hypothesis to 0.01. This value is lower than the default value of

0.05. We have chosen a smaller value of significance level with an expectation to reduce the complexity of the learned BNs as reducing the significance level results in fewer edges.

For the experiments using the KES algorithm (presented in Section 3.4) for learning structure of BNs our implementation of the algorithm written in the JAVA programming language was used. As it was noted the KES algorithm allows a trade-off between greediness and randomness by setting the value of the parameter k. Hereinafter presented experimental evaluation was done with setting $k = 1$.

As our experiments are run with real-world data noisy instances are expected. In order to avoid untruthful results computing the similarity measure between pairs of clusters represented by BNs we decided to leave some data out for eluding outliers and consider only the best representatives of the model. In the experiments presented further, computation of the similarity measure is based on 10% of instances from a cluster when the instances are ranked following the approach described in Chapter 4.

Experimental evaluation of the proposed model-based clustering algorithm using BNs is made with real-world data. Specificaly, the evaluation in real-world data applies this paper's proposal to gene expression data.

5.2 Gene expression data

Development of high throughput data acquisition technologies in biological sciences, and specificaly in genome sequencing, together with advances in digital storage and computing, have lead to the development of *DNA microarray* experiments. DNA microarray allows the monitoring and measurement of the expression levels of thousands of genes simultaneously in an organism.

Important biological knowledge is implicit in gene expression data, but extracting it is a difficult task. Therefore, there has been a significant recent interest in the development of new methods for functional interpretation of microarray gene expression datasets. Hierarchical approach for clustering biological data was found to be very useful as the hierarchy provides insight about cellular mechanisms. Genes that are similarly expressed are often involved in the same cellular processes, so that clustering suggests functional relationships between clustered genes [33].

The database involved in the evaluation is the leukemia database, which has become pretty much of a standard test base for gene expression data analysis techiques [13]. It consists of 72 samples from leukemia patients, with each sample being characterizes by the expression levels of 7129 genes. Each sample is labelled with the specific type of acute leukemia the patient suffers from: acute lymphoblastic leukemia (ALL) or acute myeloid leukemia (AML). The database consists of 47 samples labelled as ALL and 25 labelled as AML.

The preprocessing of the leukemia database was kept at minimum as possible. This means that data were only discretized. Following the most common approach in the literature, gene expression levels were discretized into three states, corresponding to the concepts of a gene being either underexpressed, or baseline, ar overexpressed with respect to its control expression level [12]. The discretization method that was used is based on information theory [16]. The resulting discretized database $D_{leukemia}$ was separated into two assisting databases where one of them contains ALL patients while the other takes in AML patients. For the sake of interpretability of the experimental results both databases were reduced to ten patients. The first database includes the ten first ALL patients in $D_{leukemia}$ while the second database contains the first ten AML patients in $D_{leukemia}$. These two newly created databases were transposed, so that the 7129 genes were the cases and the measurements for the corresponding ten patients were the predictive attributes. The resulting databases are denoted D_{ALL} and D_{AML}.

Peña et al. [29] report the experimental results that suggest the generative mechanisms underlying D_{ALL} and D_{AML} consist of three physical processes each, with each physical process being governed by a joint probability distribution that tends towards genes being underexpressed, or baseline, or overexpressed for all the patients. This finding will be the direction to follow while interpreting the experimental results presented further.

Examination of the homogeneity of the clusters underlying D_{ALL} and D_{AML} revealed that clusters underlying D_{AML} are slightly more homogeneous than those underlying D_{ALL} [29]. Therefore, the first part of the evaluation is carried out in more homogeneous D_{AML}.

5.3 Results and discussion: D_{AML}

As we specified earlier two algorithms representing different approaches will be applied for structure learning of BNs. We start from the experimental results obtained employing the PC algorithm.

5.3.1 PC algorithm

In order to get easier apprehensible experimental results we fixed the number of the initial clusters relatively small number of ten. This setting is kept the same in all following experiments.

The fusions made at each stage of the algorithm are shown in Figure 5.1. Clusters are represented by centroids of each attribute, i.e. average values of genes assigned to this cluster for all ten AML patients. Thus, each box contains ten lines which show the average level of gene expression for a certain patient. In the beginning (0 step) clusters were ordered according to the average of cluster centroids: from a cluster where genes are most likely to be unexpressed (on the

top) to a cluster with mostly overexpressed genes (on the bottom). This ordering should help keeping trace of merging.

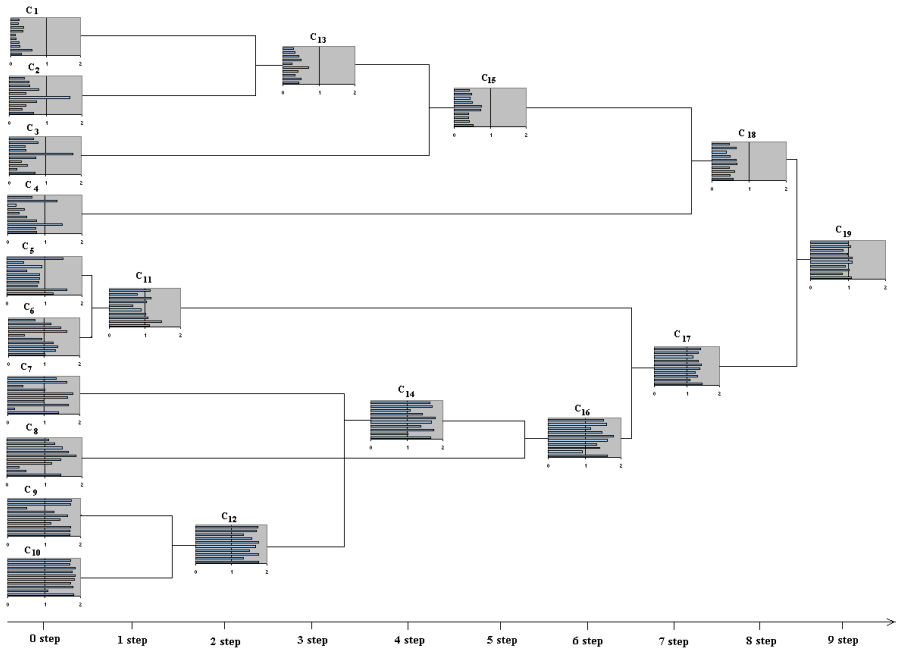


Figure 5.1: The dendogram for clustering results with the PC algorithm employed for structure learning from D_{AML}

Starting the analysis it should be noticed that the majority of initial clusters have at least one patient whose genes do not support the trend of a cluster for genes being in a certain state. For example, c_2 exhibits the tendency towards all variables being in state 0 (underexpressed) but genes of one patient show the tendency to be in state 2 (overexpressed). The same situation can be found in other clusters.

From the presented dendogram we can see that merging of clusters appears to be consequent as the clusters that are adjacent by their centroids tend to be merged. The only fusion where it can look like the rule of the closest clusters being merged first is broken (c_7 and c_{12} are merged skipping c_8) can be explained by identity of c_7 and c_8.

Our next interest is to find the most probable partition, i.e. cutting point in the dendogram. As we declared earlier stopping criteria is designed to counsel us on this issue. Figure 5.2 presents BIC score for global model at each stage of the algorithm.

Introducing the stopping criterion based on BIC score for global model we had the expected model of its performance. We expected to see a growth of BIC score in the beginning of the algorithm while fusions of clusters improve the global model and a fall of BIC score when two divergent clusters are merged.

Figure 5.2: BIC score for the global model (D_{AML}, PC algorithm)

However, analysing the visualization of BIC score we can see only a decrease in score after each merging is done. This can be explained by the high complexity of models learnt by the PC algorithm. Thus, we will look for the biggest drop in BIC score and consider it as a suggestion that the most probable partition is found.

From the figure we can see two falls in BIC score: going from six to five clusters and from four to three clusters. However, the second drop in score is much more significant so we can say that the stopping criterion advises that four clusters is the most probable partition.

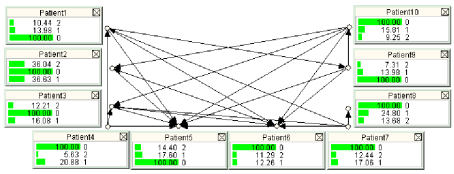
Being adviced by the stopping criterion we can come back to Figure 5.1 and analyse the four final clusters. We learn that there are three compound clusters presenting different states of variables: c_{15} (includes three initial clusters) shows a tendency towards variables being in state 0, c_{11} (includes two initial clusters) towards being in state 1 and c_{16} (includes four initial clusters) towards being in state 2. Additionaly, we find the cluster c_4 that was left aside and was not merged with the other clusters. Four clusters we find as final ones are quite different in their sizes, i.e. number of instances included. See Table 5.1.

Cluster	Number of instances
4	583
11	1311
15	2313
16	2922

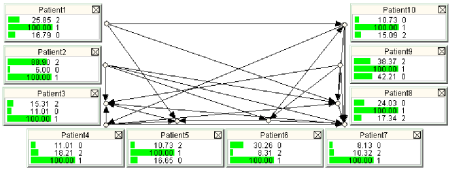
Table 5.1: Size of final clusters

BNs representing the four final clusters are shown in Figure 5.3. In order to see the tendencies variables being in a certain state max propagation was performed. This operation computes the most probable configuration of all variables. This

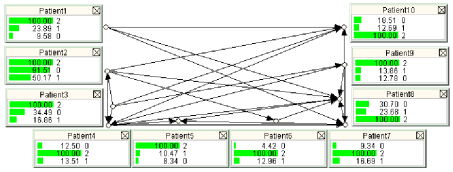
is computed from the joint probability distribution.



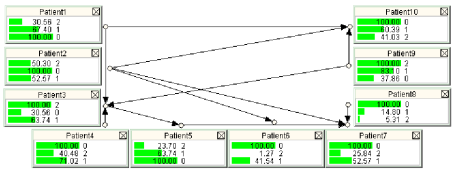
(a) Cluster 15



(b) Cluster 11



(c) Cluster 16



(d) Cluster 4

Figure 5.3: BNs learned from the clusters (D_{AML}, PC algorithm)

The figure suggests that in the cluster 15 the most probable state for all ten patients is 0, in the cluster 11 the most probable state for all variables is 1 and in the cluster 16 the most probable state is 2. However, we cannot find such a convincing description for the cluster 4 where eight patients tend to have unexpressed genes while two patients tend to have the same genes overexpressed.

Further the analysis of the results gotten employing the KES algorithm are presented.

5.3.2 KES algorithm

The fusions made at each stage of the proposed model-based algorithm are shown in Figure 5.4. It should be reminded that because of the initialization method for k-means algorithm initial clusters are different from the ones in the previous run with the PC algorithm applied.

The presented dendogram suggests that merging of clusters is sensible as the clusters that are adjoining according to their centroids tend to be merged. One exception can be found: cluster 8 and cluster 10 are fused ignoring the cluster 9. However, these clusters are very close to each other if the comparison is based on the values of centroids.

BIC score for the global model at each stage of the algorithm is presented in Figure 5.5.

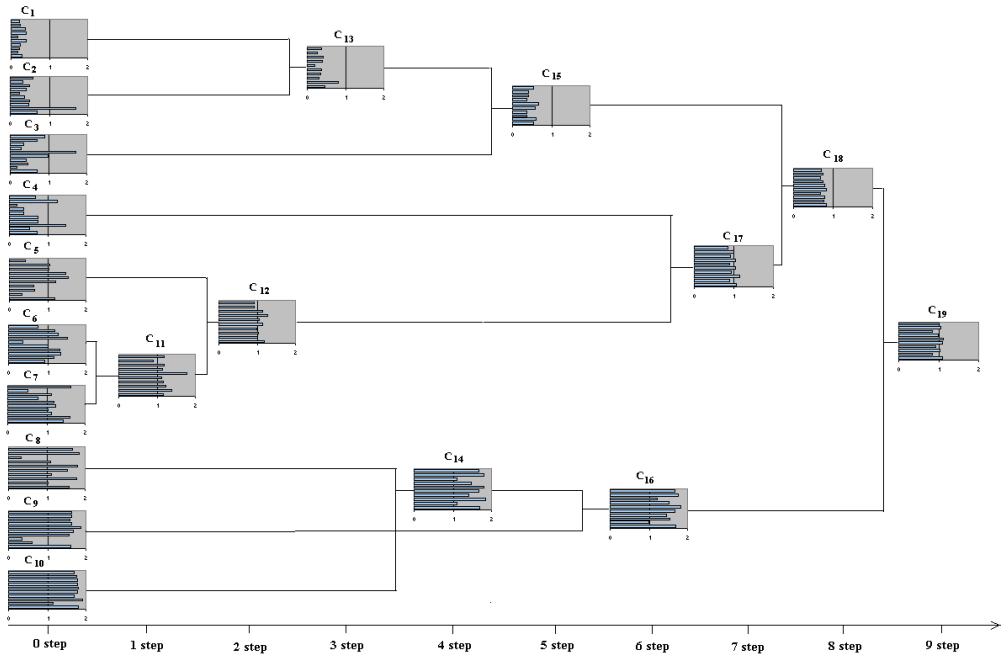


Figure 5.4: Dendogram for clustering results with the KES algorithm employed for structure learning of BNs from D_{AML}

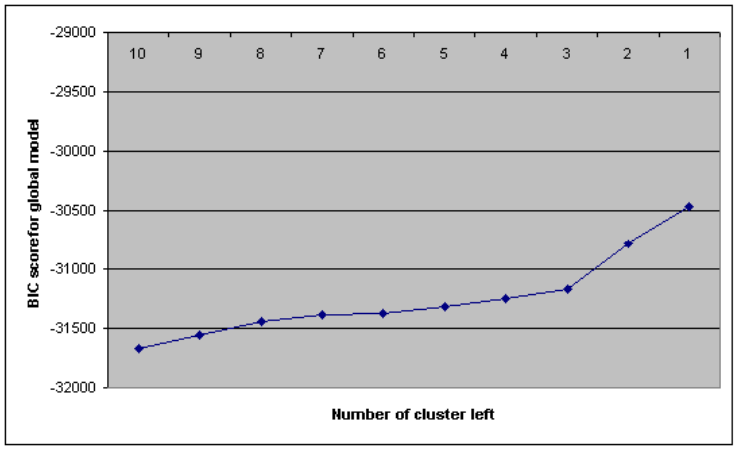


Figure 5.5: BIC score for the global model (D_{AML}, KES algorithm)

In the evaluation of the experimental results with the PC algorithm employed the biggest fall in BIC score for the global model was considered to determine the most probable partition. However, such a determination cannot be applied for the results with the KES algorithm as BIC score for the global model increases at each stage. This is closer to our expectations that the score should grow in the

beginning. However, there is no fall in BIC score now. The possible interpretation of such performance is the same structure shared by all generative mechanisms underlying gene expression data we use.

One possible way to find the most probable partition is to consider the same number of final clusters as in the results with the PC algorithm applied, i.e. four clusters. If we choose four clusters as the most probable partition we obtain three composed clusters with the tendency towards all variables being in one state and one initial cluster - cluster 4. This cluster that was not included in the composed clusters is very similar to the cluster 4 from the experiments with the PC algorithm.

As the cluster left aside from merging is almost the same as in the previous results we try to analyse the results when the most probable model includes three clusters. Thus, we can check if this partition contains three clearly distinct physical processes. Table 5.2 presents the sizes of three final clusters.

Cluster	Number of instances
15	2178
16	2305
17	2646

Table 5.2: Size of final clusters

Figure 5.6 presents BNs representing the clusters 15, 16 and 17. The graphs after performing max propagation show that genes tend to be underexpressed for all patients in cluster 15, baseline for all patients in cluster 17 and in cluster 16 nine patients have mostly overexpressed genes while one patient has the same genes underexpressed.

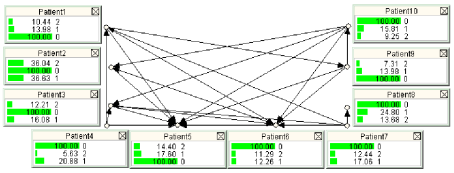
5.4 Results and discussion: D_{ALL}

We continue with the experimental evaluation in D_{ALL}.

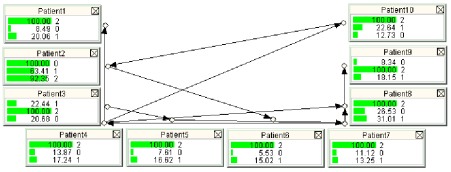
5.4.1 PC results

The dendogram for the results obtained running the model-based hierarchical clustering algorithm with the PC algorithm employed are shown in Figure 5.7. This visualization of merging process confirms that the proposed algorithm performs in the sensible way.

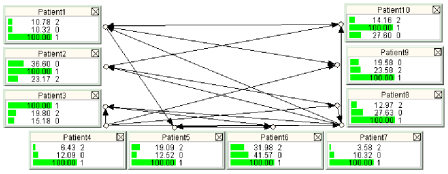
BIC score for the global model shown in Figure 5.8 suggests that the most probable number of clusters is six. The curve for BIC is rather convincing as reducing the number of clusters from six to five causes the drop of BIC from -35140 to -123709. Following the advice of the stopping criteria we continue with analysis of six clusters: c_6, c_7, c_8, c_{11}, c_{13} and c_{14}. The sizes of these clusters



(a) Cluster 15



(b) Cluster 16



(c) Cluster 17

Figure 5.6: BNs learned from the clusters (D_{AML}, KES algorithm)

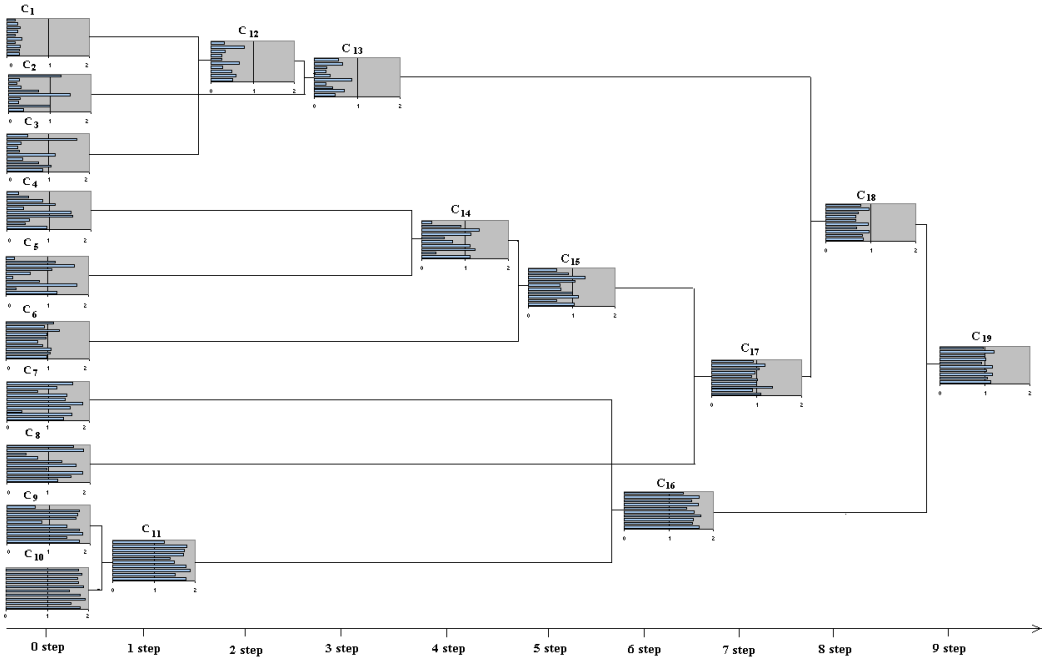


Figure 5.7: Dendogram for clustering results with the PC algorithm employed for structure learning of BNs from D_{ALL}

are given in Table 5.3 and directed graphical models learned from the clusters are shown in Figure 5.9.

Figure 5.8: BIC score for the global model (D_{ALL}, PC algorithm)

Cluster	Number of instances
6	850
7	502
8	793
11	1937
13	2000
14	1047

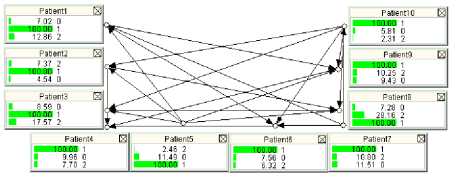
Table 5.3: Size of final clusters

Three clusters from the six represented are combined clusters. The cluster 11 contains the genes that show the tendency to be overexpressed. Genes in the cluster 13 tend to be underexpressed, so do the genes from the cluster 14. However, the latter cluster has three patients with mostly overexpressed genes. The cluster 6 includes genes that tend to be baseline, the same tendency just with four exceptions corresponds to the cluster 8. Finally, the cluster 7 contains genes that are overexpressed for the majority of variables.

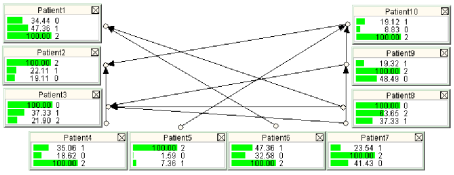
5.4.2 KES results

Figure 5.10 presents the obtained hierarchy of clusters running the algorithm when the KES algorithm is applied.

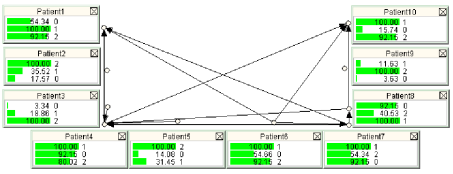
The curve of BIC score for the global model in Figure 5.11 shows the slight increase after each merging. As this does not give us a hint about the number of the underlying generative mechanisms we will assume three of them to exist as it was suggested by experiments of other researchers [29].



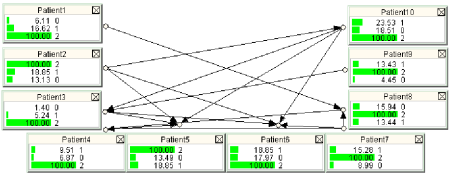
(a) Cluster 6



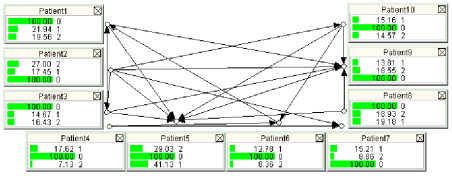
(b) Cluster 7



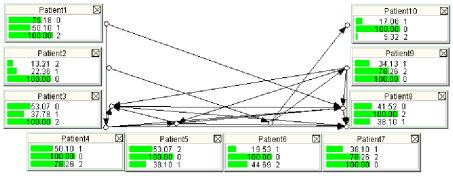
(c) Cluster 8



(d) Cluster 11



(e) Cluster 13



(f) Cluster 14

Figure 5.9: BNs learned from the clusters (D_{ALL}, PC algorithm)

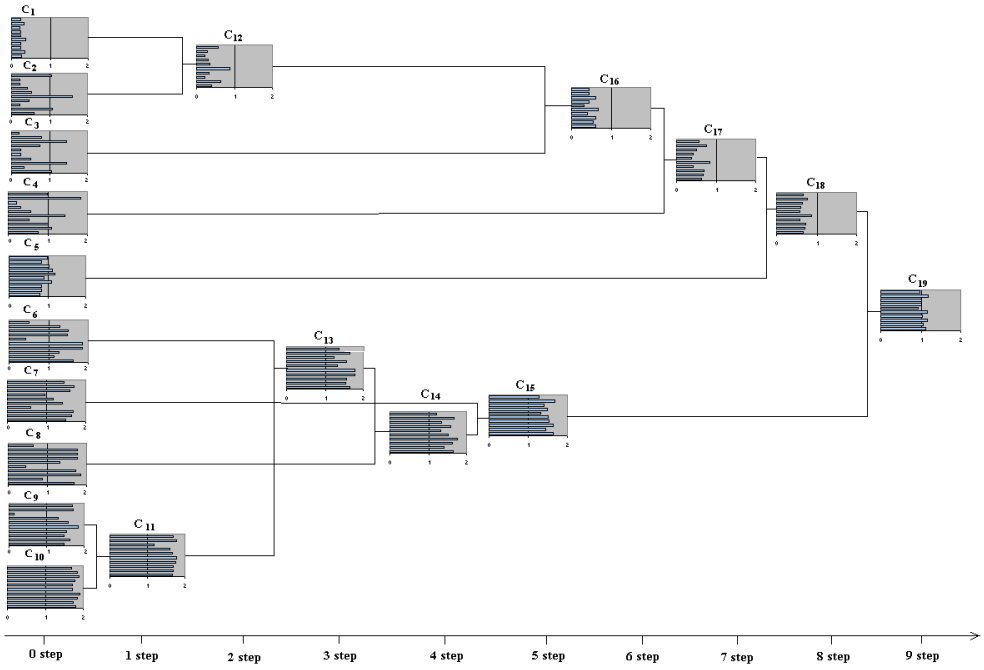


Figure 5.10: Dendogram for clustering results with the KES algorithm employed for structure learning of BNs from D_{ALL}

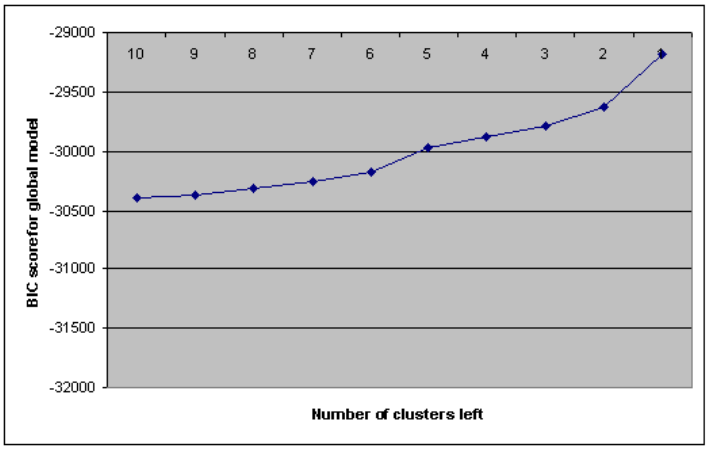


Figure 5.11: BIC score for the global model (D_{ALL}, KES algorithm)

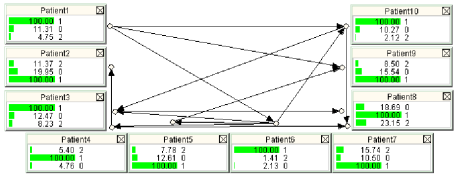
From Table 5.4 we can see that two of final clusters are big clusters while one of them, cluster 5, is relatively small.

Two of three final clusters shown in Figure 5.12 are combined clusters where the cluster 17 contains genes with the tendency to be underexpressed while the cluster 15 includes genes that tend to be overexpressed. Genes that tend to be

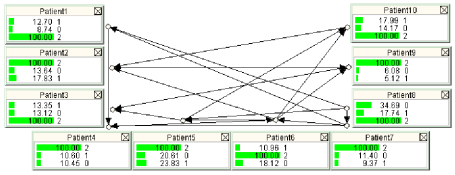
Cluster	Number of instances
5	879
15	3405
17	2845

Table 5.4: Size of final clusters

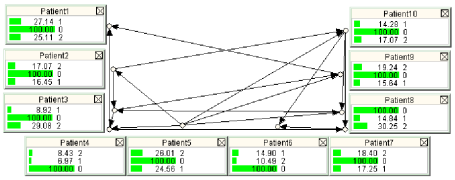
baseline are collected in cluster 5 that was not fused with other clusters. The nice thing to notice is that the tendencies of genes to be in a certain state are supported by all variables in all three graphical models.



(a) Cluster 5



(b) Cluster 15



(c) Cluster 17

Figure 5.12: BNs learned from the clusters (D_{ALL}, KES algorithm)

5.5 Summary

The experimental evaluation showed that the model-based hierarchical clustering algorithm we proposed performs in a sensible manner and the similarity measure for merging the clusters represented by BNs is proper for this task.

The other issue is stopping criteria that was designed to advice the most probable model. We have obtained diverse results with different algorithms for structure learning of BNs being employed. Having PC algorithm applied BIC score decreases all the time and at some point it has a significant drop thus indicating the most probable partition. On the other hand, BIC score slightly increases all the time running the algorithm with application of the KES algorithm.

The possible explanation for such results could be different dimensionality of BNs learned by these two structure learning algorithms. It was found that BNs learned by KES algorithm did not have more than three parents for one node. This can be seen as a bound on the maximum number of parents of each node. However, it is known that in real genetic networks some genes can have very high fan-in and fan-out [22]. Therefore, the PC algorithm which characteristics do not create such a tight bound for the number of parents tends to learn BNs with high dimensionality. BIC score rapidly decreases because of the growing dimensionality of the learned BNs and high penalty for this.

The fact that BIC score for the clustering algorithm using the KES algorithm never decreases can be explained by the same structure shared by all generative mechanisms underlying gene expression data we use.

6 Conclusions

The main contribution of this paper has been the proposal and experimental evaluation of the model-based clustering algorithm which represents clusters as BNs and adopts hierarchical clustering strategy. The evaluation has been done on gene expression databases as the physical processes underlying this data could be assumed on the basis of the investigations made by other researchers this way simplifying the task of results analysis.

The model-based hierarchical clustering algorithm creates a dendogram where each cluster is represented by a directed graphical model that enables to exploit the advantages of BN paradigm over other paradigms such as evidence propagation and ability to read conditional independencies.

Two model structure learning algorithms representing different approaches for structure learning, the PC algorithm and the KES algorithm, were applied. Regardless of diverse results in the dimensionality of learned BNs, both algorithms showed the same tendency to merge the nearest graphical models. Therefore, the results that have been reported in this paper show the sensemaking performance of the introduced model-based hierarchical clustering algorithm.

We have also analysed the stopping criterion which should be a helpful tool advicing on the best model to select from the sequence of nested partitions. The preliminary results proved the assistance of it in selecting the best partition. However, additional investigation is needed as gene expression data used in empirical evaluation turned out to be pretty much unsuitable for testing the introduced stopping criterion.

7 Future work

The experimental results reported in this study can be seen as a preliminary stage for further investigation on the introduced model-based hierarchical clustering algorithm. We claim so because of the specifics of the data that has been used for the empirical evaluation of the approach. Future research could be done with data that is known to have several generative mechanisms with different structure underlying the data. Such a study could help to learn more about the properties of the introduced stopping criterion.

Study of other possible similarity measures between two clusters represented by BNs that would guide the model-based hierarchical clustering algorithm in merging is another possible line for future research.

Bibliography

[1] M. Aitkin, D. Anderson and J. Hinde. *Statistical Modelling of Data on Teaching Styles*. Journal of the Royal Statistical Society, 144:419-461, 1981.

[2] J. D. Banfield and A. E. Raftery. *Model-based Gaussian and non-Gaussian clustering*. Biometrics, 49:803-821, 1993.

[3] P. Cheeseman and J. Stutz. *Bayessian classification (AutoClass): Theory and results*. In Advances in knowledge discovery and data mining, 153-180, 1995.

[4] D. M. Chickering. *Learning Bayesian Networks is NP-Complete*. Learning from Data: Artificial Intelligence and Statistics, 121-130, 1996.

[5] D. M. Chickering. *Optimal Structure Identification with Greedy Search*. Journal of Machine Learning Research, 3:507-554, 2002.

[6] G. Cooper and E. Herskovits. *A Bayesian method for the induction of probabilistic networks from data*. Machine Learning, 9:309-347, 1992.

[7] A. P. Dempster, N. M. Laird and D. B. Rubin. *Maximum Likelihood from Incomplete Data via the EM Algorithm*. Journal of the Royal Statistical Society, 39:1-38, 1977.

[8] R. O. Duda, P. E. Hart. *Pattern Classification and Scene analysis*. John Wiley and Sons, 1973.

[9] B. S. Everitt. *Cluster Analysis - Third Edition*. Arnold, 1993.

[10] C. Fraley. *Algorithms for model-based Gaussian hierarchical clustering*. SIAM Journal on Scientific Computing, 20(1):270-281, 2000.

[11] W. Frawley, G. Piatetsky-Shapiro and C. Matheus. Knowledge Discovery in Databases: An Overview. AI Magazine, 213-228, 1992.

[12] N. Friedman, M. Linial, I. Nachman and D. Pe'er. *Using Bayesian Networks to Analyze Expression Data*. Journal of Computational Biology, 7:601-620, 2000.

[13] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield and E. S. Lander. *Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring*. Science, 286:531-537, 1999.

[14] J. Han and M. Kamber. *Data Mining, Concepts and Techniques*. Academic Press, 2001.

[15] http://www.hugin.com.

[16] I. Inza, P. Larrañaga, R. Blanco and A. J. Cerrolaza. *Filter Versus Wrapper Gene Selection Approaches in DNA Microarray Domains*. Journal of Artificial Intelligence in Medicine, 2003.

[17] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer, 2001.

[18] R. Jurgelenaite, A. O. Mohamed, N. Søndberg-Madsen and C. Thomsen. *Industrial Datamining*. Report on first half of Master Thesis, 2003.

[19] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data*. John Wiley & Sons, Inc., 1990.

[20] J. A. Lozano. *Algoritmos Genéticos Aplicados a la Clasificación No Supervisada*. Ph.D. Thesis, University of the Basque Country, 1998.

[21] G. W. Milligan. *An examination of the effect of six types of error pertubation on fifteen clustering algorithms*. Psychometrica, 45:325-342, 1980.

[22] K. Murphy and S. Mian. *Modelling Gene Expression Data using Dynamic Bayesian Networks*. Technical report, 1999.

[23] K. Murphy. *A Brief Introduction to Graphical Models and Bayesian Networks, 2003*. http://www.ai.mit.edu/ murphyk/Bayes/bayes.html.

[24] J. D. Nielsen, T. Kočka and J. M. Peña. *On Local Optima in Learning Bayesian Networks*. In Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence, 2003.

[25] B. Palace. Data mining: What is data mining?, 1996. http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.htm.

[26] J. M. Peña, J. A. Lozano and P. Larrañaga. An empirical comparison of four initialization methods for the K-Means algorithm Pattern Recognition Letters, 20:1027-1040, 1999.

[27] J. M. Peña. *On Unsupervised Learning of Bayesian Networks and Conditional Gaussian Networks*. Ph.D. Thesis, University of the Basque Country, 2001.

[28] J. M. Peña, J. A. Lozano and P. Larrañaga. *Unsupervised Learning of Bayesian Networks Via Estimation of Distribution Algorithms*. In Proceedings of the First European Workshop on Probabilistic Graphical Models, 144-151, 2002.

[29] J. M. Peña, J. A. Lozano and P. Larrañaga. *Unsupervised Learning of Bayesian Networks Via Estimation of Distribution Algorithms: An Application to Gene Axpression Data Clustering*. submitted, 2003.

[30] M. Ramoni and P. Sebastiani. *Bayesian Methods*. Intelligent Data Analysis: an Introduction. Springer, 129-166, 2002.

[31] M. Ramoni, P. Sebastiani and P. Cohen. *Bayesian clustering by dynamics*. Machine learning, 47:91-121, 2002.

[32] A. Rauber, J. Paralic and E. Pampalk. *Empirical Evaluation of Clustering Algorithms*. Journal of Information and Organizational Sciences, 24:195-209, 2000.

[33] E. Segal and D. Koller. *Probabilistic Hierarchical Clustering for Biological Data*. In Proceedings of the Sixth International Conference on Research in Computational Molecular Biology, Washington, 2002.

[34] S. Z. Selim, M. A. Ismail. *K-Means-type algorithms: a generalized convergence theorem and characterization of local optimality*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 6:81-87, 1984.

[35] M. Singh, M. Valtorta. *Construction of Bayesian Network structures from Data: a Brief Survey and an Efficient Algorithm.* International Journal of Approximate Reasoning, 12:111-131, 1995.

[36] G. Schwarz. *Estimating the Dimension of a Model*. The Annals of Statistics, 6:461-464, 1978.

[37] P. Spirtes and C. Glymour. *An algorithm for fast recovery of sparse causal graphs*. Social Science Computer Review, 9(1):62-73, 1991.

[38] S. Vaithyanathan and B. Dom. *Model-based hierarchical clustering*. The Sixteenth Conference on Uncertainty in Artificial Intelligence, 2000.

[39] J. H. Ward. *Hierarchical grouping to optimize an objective function*. Journal of the American Statistical Association 58:236-244, 1963.