Parallel Clustering Algorithms for Categorical and Mixed Data

Nguyen Thi Minh Hai

Advisor: Professor Horiguchi Susumu

2-2004

Multi-Media Systems Laboratory

Japan Advanced Institute of Science and Technology

Abstract

Clustering is a fundamental and important technique in many research fields such as image processing, pattern recognition, machine learning, etc. Since data to be processed are rapidly growing larger. The clustering is very important to mine and to find meaningful knowledge among huge data sets. In addition, the numeric data and categorical data are popular in recent databases and it is difficult to cluster database which consists of both these two kind of data. Thus, improving a good clustering algorithm for widely applicable is strongly expected. In addition, the data is too large to be analyzed in acceptable time by using serial clustering algorithm, even scalable clustering algorithm. Also, the huge size of data leads the fact that the data can not be stored in single memory. Consequently, efficient and efficiency parallel clustering algorithms are hopefully will be proposed in order to deal with both complex data and huge data.

In this thesis, we propose a new serial clustering algorithm which can achieve high accuracy result when perform with categorical and mixed data. The algorithm named at High Accuracy Clustering Algorithm for Categorical and Mixed Data based on a new Grouping Approach (HAC algorithm). The algorithm was improved from k-sets, a new clustering algorithm which can get better results than the previous algorithms when doing the same task on the same databases of their experiments. Our proposed algorithm always achieves the highest accuracy compare with the previous researches on the same tests. Furthermore, the HAC algorithm can overcome the problem of local optimization of the various previous researches, as well as, do not depend on the order of input data.

An effective and efficient clustering algorithm is the one has to be able to deal with complex and huge databases. Parallelization is one useful tool which helps the clustering algorithm can save much processing time. In the research, we also proposed two parallel algorithms which parallelize HAC algorithm by different ways. Both of our parallel algorithms can reduce significantly time in processing with huge databases and also get high speed-up in large enough number of processors.

In summary, we proposed a nice serial clustering algorithm which can get high accuracy results and has some other advantages which did not appear in the previous researches. In additions, our proposed parallel clustering algorithms are effective and are able to deal with complex-huge-databases.

Acknowledgements

First of all, I would like to express my deepest thanks to my advisor, Professor Susumu Horiguchi. It was my good fortune to have he as my supervisor during my master course at Japan Advanced Institute of Science and Technology (JAIST). He taught me not only scientific knowledge but also the good living style. His thorough scientific approach and unending quest for excellence, and furthermore, his encouragements at the right time, his kindly treats have been inspirational in the years of my thesis research. I have learnt many things from him. However, I wish I had listened to his advice more often.

I would like to express my great thanks to Professor Ho Tu Bao of JAIST for his invaluable guidance, support, and encouragement. His constructive criticism and useful discussions improved my understanding of finding research problem and clustering knowledge.

I am grateful to my associate researchers of my lab, Dr. Ryoko Hayashi and Dr. Masaru Fukushi for their sharing valuable experiences, numerous suggestion through the years of mine in the lab.

I sincerely thank all my friend and colleagues who always support me in time of need. I greatly appreciate to my lab-mates for their contributions in making a wonderful and supportive academic environment. Specially, Associate Professor Yasushi Inoguchi, Dr. Hiroshi Horii, Mr. Yoshiki Yazawa, and Miss. Eeiko Sugawara – my tutor - gave me many helps and encouragements through the years. I like to give the special thanking to Mr. Le Si Quang and Mr. Nguyen Phu Chien for their helpful discussion and good friendship to make me overcome many troubles. The life would be nothing without friend. By the way, I want to send my heartfelt thanks to all of my friends in Jaist and in other places.

I am deeply indebted to the Okazaki Kaheita International Scholarship Foundation for granting me a scholarship and a chance to come to Japan. Without that, I couldn't come to study and enjoy like in Japan. I also would like to thanks to the people at the Okazaki Kaheita International Scholarship Foundation. They brought me cheerful spirit to spend my life in Japan.

I am happy to be able to express my deep thanks to Mrs. Etsuko Horiguchi for her warm hearted treats which encouraged me very much during my stay in JAIST. She is not only my Japanese teacher but also my "great-friend". She taught me many things about Japan, Japanese and gave me

the optimism, the self-confidence when I had trouble. Without her, it is very difficult for me to enjoy life here.

Finally, I have saved the best for the last. I wish to express my endless love and gratitude to my family for always being there when I needed them and supporting me through all my life. I am especially grateful to my parents for everything they taught me and for all the sacrifices they made for me. Lastly, I cannot help giving the thanks to my sweetheart for all the sentiment, encouragement, help and sympathization he gave me.

Thank you very much for all!!!

Table of Content

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Introduction and Motivations

Clustering is an important area of application for variety of fields including data mining [7], statistical data analysis [33], machine learning [36], pattern recognition, and image processing [10], etc. Cluster analysis divides unlabeled data into groups of similar objects to achieve simplification in managing data as well as mining useful knowledge from data. The categorical data or the data consists of both numeric and categorical are popular in recent databases. The key point of clustering process is to determine similarity measures between data objects. The well-known similarity measures for categorical data are calculated by the ratio of number of variables for which two categorical objects are the same state to the number of total variables [19]. This may lead to the failing of determine similar objects in case of two objects which are different from each other but have the same distance to the third object because of having the same number of values which have the same state to the third object. However, it is unable to perform mathematical operations on the categorical data. Therefore, building effective similarity measure for categorical data, especially for data consists of both categorical and numeric data is still a big challenge.

Recent developments of clustering methods for categorical data can be observed in two groups. One focuses on extension of the *k*-means algorithm [35], such as the transformation of categorical data [42], *k*-modes algorithm [21], *k*-prototypes algorithm [22], fuzzy *k*-modes algorithm [23], *k*-modes algorithm with tabu search [38] and the fuzzy k-modes algorithm with tabu search [45]. The other group follows different principles, typically the methods STIRR [13], CACTUS [12], ROCK [17].

In [42] the author converted multiple categorical attributes into binary ones, and treat the binary attributes as numeric by the *k*-means algorithm. The main drawback of the approach is that the cluster means, given by values between 0 and 1, often do not indicate the exact characteristics of the clusters. Huang [21] proposed the k-modes algorithm based on the k-means paradigm to cluster categorical data by using modes instead of means for clusters. Huang and Ng [23], [45] also proposed a fuzzy k-modes algorithm by extending the fuzzy k-means. However, these two algorithms are unstable due to the non-uniqueness of the modes. That is, clustering results depend on selections of the modes. To solve this problem, Ng and Wong [38], [43] extended the k-modes algorithm with the Tabu search technique. Sun et al. [45], [27] proposed enhancement strategies for k-modes and fuzzy k-modes algorithms using an iterative initial-point refinement. Although these techniques can improve the quality of clusters, they are very time-consuming.

STIRR [13] is an iterative method, investigated in terms of certain types of non-linear dynamical systems, for assigning and propagating weights on categorical values in a table. STIRR highly depends on the choice of its combining operator, and produces clusters that might require a heavy post-processing stage. CACTUS [12] is an approach that entails the computation of summaries from the underlying data is sufficient to compute a set of "candidate" clusters which can then be validated to determine the actual set of clusters. The disadvantage of this algorithm is that running time of CACTUS is growing so fast when the number of dimensions grows. ROCK [17] is a hierarchical algorithm that uses a new concept of links—computed as the number of common neighbors between two objects—to measure the similarity/proximity between a pair of data objects. ROCK can discover good quality clusters but its running time is still quite high.

Recently, Le and Ho [34] proposed a clustering method for categorical and mixed data, called the k-sets algorithm for categorical and mixed data. Unlike the other methods, this algorithm does not need to select the initial points (modes) of the clusters. Therefore, the results are stable. In addition, it was shown in [34] that the clustering accuracy is much better than those of previous clustering algorithms [21], [22], [23], [38], [45] when the performance was evaluated on the same databases collected from the UCI repository. However, it is not clear whether the k-sets algorithm can achieve high clustering quality for huge.

It is useful to see that clustering techniques find interesting and previously unknown patterns in large scale data, embedded in a large multi-dimensional space and are applied to a wide variety of problems like customer segmentation based on similarity of buying interest [47], detection of clusters in geographic information systems [39], etc. Then, clustering algorithms need to efficiently scale up with the multi-dimensional of data sets and also the huge size of data. The aforementioned sequential clustering algorithms have a common problem that prevents them from running in a reasonable time when applied to huge databases on serial computers. To deal with huge databases, the people have tried to develop algorithms with small processing times. However, there still are clustering algorithms which can not possibly deal with a very large database in an acceptable time. Therefore it is necessary to develop parallel algorithms that can deal with very huge databases. Sanpawat Kantabutra et al. [30]and K. Stoffel et al. [43]proposed parallel clustering algorithms based on the k-means clustering algorithm. Their parallel k-means algorithms, however, are only to numeric databases only. Also, most other parallel clustering algorithms can deal with the databases consist of numeric attributes only [9], [27].

The above reasons encourage us to develop parallel clustering algorithms for categorical and mixed data. In this thesis, we first concentrate on improvement of the k-sets clustering algorithm to achieve higher accuracy. It is then followed by parallel versions of the improved algorithm. The proposed parallel algorithms are expected to be scalable for both very large databases and for complex data.

1.2 Contribution of the Thesis

In this thesis, we have proposed a new serial clustering algorithm called High Accuracy Clustering Algorithm for Categorical and Mixed Data based on a new Grouping Approach (HAC algorithm) and two parallel algorithms which parallelized the HAC algorithm. The HAC algorithm was proposed based on the improving approach for the re-partition step of k-sets to aim at achieving higher quality in clustering results.

The contributions of this thesis can be summarized as follows:

- Proposed an approach to improve the re-partition step of k-sets to achieve high clustering accuracy. The new algorithm is HAC algorithm.

- Analysis the complexity of the HAC algorithm and compare the performance with other methods on the same databases. The comparison shows that our new approach always achieve highest accuracy in clustering results.
- To enable HAC algorithm to deal with huge databases, we proposed a parallel approach, which concentrate on parallelize the trial of one input parameter. This input parameter is the threshold to determine if two data objects are considered as similar to each other or not.
- In case of mixed data, we consider to employ multi-level parallel techniques to parallelize on more than two input parameters. This is hopefully to effectiveness in case of large databases and large number of processors.
- Furthermore, the performance of our two parallel algorithms are evaluated and analysis.
- Last is our analysis of the advantages and disadvantages of our research and the future plan to overcome drawbacks.

1.3 Organization

After the introduction chapter, the rest of the thesis is organized as follows:

- As our research aimed at proposing both serial and parallel clustering algorithms for categorical and mixed data, therefore, it is important to take a overview on clustering tasks, clustering techniques, parallel techniques as well as the researches of clustering tasks, parallel clustering in general but emphasize on researches and techniques closely related with our research. All will be presented in chapter 2.
- The chapter 3 is starting point for our contributions on the thesis. In chapter 3, after having a general view at clustering for categorical and mixed data, and the development as well as relations of parallel techniques with

clustering; and before of proposing a new clustering algorithm based on the approach of improving the re-partitioning step in k-sets algorithm; it is suitable and necessary to introduce in detail the k-sets clustering algorithm as well as its computational complexity, it advantages and disadvantages, etc. Then, we express the reasons for our way to improve the k-sets. Following is the detail of our proposed algorithm named High Accuracy Clustering Algorithm for Categorical and Mixed Data based on a new Grouping Approach (HAC algorithm) and its complexity analysis. At the last of the chapter 3, evaluation of both the k-sets and our proposed one will be shown and discussed in detail.

- In Chapter 4, we describe and discuss on our two propose clustering algorithms for HAC algorithm. The detail of reasons to do such parallel algorithms and how they are constructed, how they are work, how to make they work well will be discussed in detail. We also analyze the computational complexity of these two algorithms.
- Chapter 5 will shows performances of our proposed parallel algorithms in Chapter 4. The analysis of their average execution time, speed up, efficiency, and also the discussion of why we got such performance. In addition, to see more clearly the difference of the two parallel algorithms via their performances, we will take the comparisons.
- In the last chapter, we present conclusions and future works of our research. In this chapter, the summary of our thesis, the more discuss on our research’s advantages and disadvantages will be described in the conclusion section. Then, the future works will take a look again on our research and also the other researches to draw our future plans.

Chapter 2. Related Research

2.1 Introduction

Nowadays, by using good technical equipment, we can get lots of useful data from the real life. Once hold such huge data, we like to discover which information the data can tell to us. It is the fact that the database need to analysis is very huge. But we cannot analysis directly from such large data. So that, we have to use clustering technique to partition databases to the small enough groups to analysis.

Also, because of the mount of database is very huge, and in some application like diagnose, the waiting time is not good. It makes the aim to develop fast and accuracy clustering algorithm. Recently, lots of clustering algorithms are developed but they still have some drawback such as time-confusing, cannot get high accurate result, etc… Consequently, it is important to learn and develop techniques for scaling up clustering algorithms.

2.2 Data Mining

Data mining is the field which clustering plays an important task. In this section, a brief introduction on data mining will be showed.

2.2.1 Concept of Data Mining

Data mining is the process of discovering interesting knowledge from large amounts of data stored either in databases, data warehouses, or other information repositories [19]. Data mining tools predict future trends and behaviors, allowing business to make proactive, knowledge-driven decisions. Data mining tools can answer business questions that traditionally were too consuming to resolve. The tools scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations.

The improving of sciences technology help people can collect and store the databases with very large and rapidly growing larger size. Data mining techniques can be implemented rapidly to enhance the value of existing information resources. When implemented on high performance computing, data mining tools can analyze massive databases to deliver answers to questions such as, “Which clients are most likely to respond to my next promotional mailing, and why?” [46].

The following section will give a look at popular data mining techniques.

2.2.2 Data Mining Techniques

Classification: the task is to learn to assign data objects to predefined classes. Classification requires supervised learning, i.e. the training data has to specify what we are trying to learn (the classes).

Clustering: is unsupervised learning task. No predefined classification is required. The task is to learn classification from the database.

Association Analysis: is the discovery of association rules showing attribute-value conditions that occur frequently together on a given set of data.

In this thesis, we concentrate on clustering algorithm for categorical and mixed data. The related problems will be described in the following sections of this chapter.

2.3 Clustering

2.3.1 Clustering Analysis

Clustering is a fundamental technique of unsupervised learning in pattern recognition, machine learning, etc. The techniques of clustering algorithms are quite important and were discussed widely [3], [15]. Image that you are given a database of data for analysis, where, the label in database is not yet known, what should you do? In this case, it is needed to find the label of each data objects, i.e., find the relationship between data objects to help analytic task become easier. Let’s turn to do cluster to solve the problem. Clustering is used to group a set of unlabeled data objects into smaller groups of similar objects with label to achieve simplification in managing data as well as mining useful knowledge from data. Each group with the label, called a cluster, is a collection of data objects satisfying the condition that objects in this cluster are similar to one another and objects in this cluster are dissimilar to the objects in other clusters according to some defined criteria. There are various methods to process clustering task. For example, to partition objects, similar measures were used in statistical clustering methods; whereas, the conceptual clustering methods cluster objects according to the concepts objects carry, etc.

2.3.1.1 Example of Database in Clustering Task

The table 1 is an example of data representation in clustering task.

From the second row, each row is an object (or tuple). Each object consists of 4 attributes (respect with the first four columns). Each value in one box is an attribute value. The last column is class names which objects belong to. If the data consists of class names of objects, it means the database is testing data, on the contrary it is training data.

Name	Age	Income (Yen)	Job	Class
Nguyen	26	120000	Student	Usually buying book
Tran	26	50000	Student	Not usually buying book
Ly	30	200000	Engineer	Not usually buying book
Le	35	200000	Prof	Usually buying book

Table 1. Example of data representation in clustering task

2.3.1.2 Example of Data Clustering

Figure 1 is an example for clustering task in two-dimension Cartesian coordinates. Each object is a point. For this example, the objects are considered similar than the others mean that their Euclidean distance is smaller than the others.

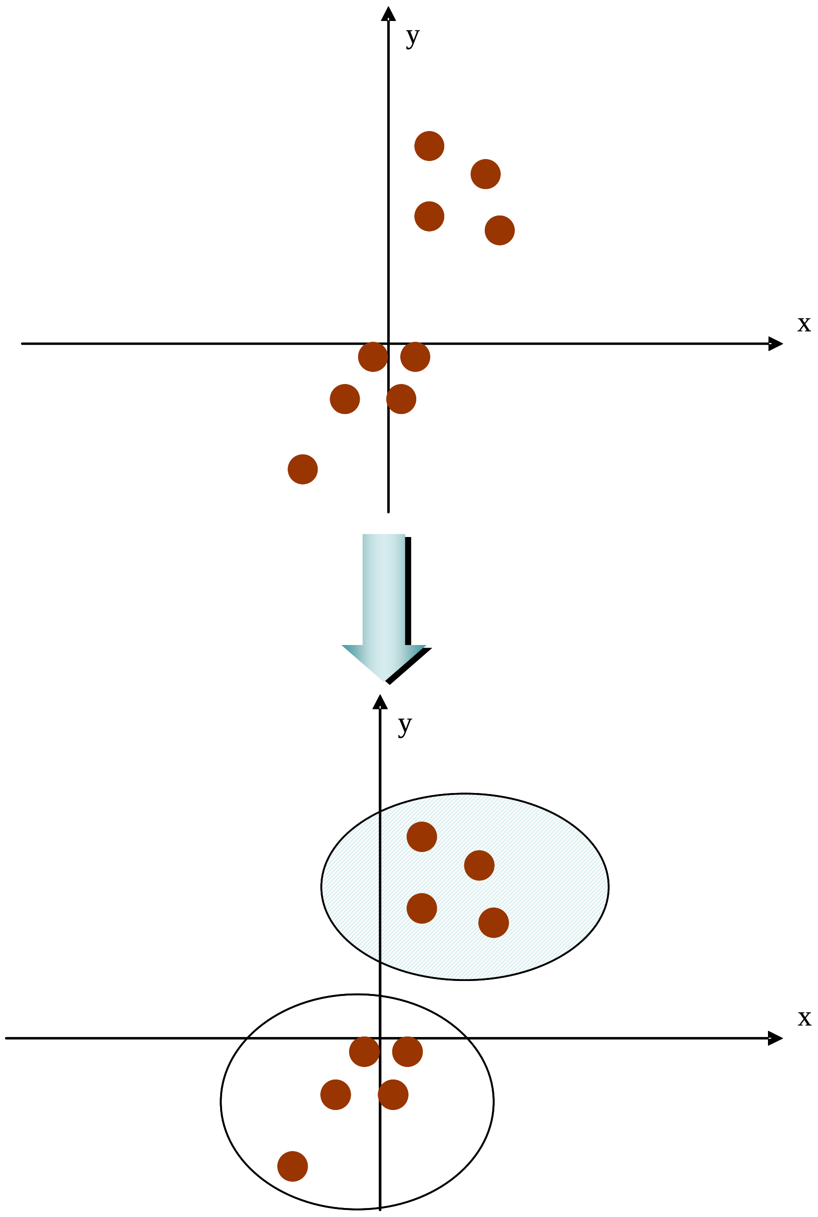

Figure 1: Example of Cluster

2.3.1.3 *Applications of Clustering*

Clustering task is a main task of data mining, which have many applications. Such as:

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Pattern Recognition
- Spatial Data Analysis
    - create thematic maps in GIS by clustering feature spaces
    - detect spatial clusters and explain them in spatial data mining
- Image Processing
- WWW
    - Document classification
    - Cluster Weblog data to discover groups of similar access patterns

2.3.1.4 *Important Criteria of Clustering Task*

Clustering is the task which tried to find structure of databases depend on the similarity between attribute values. There are some important properties of clustering algorithms we are concerned with in data mining [19], [7], [11]:

- **Efficiency and Scalability to large databases:** Recently, the datasets with thousands, then thousands or more of data objects are popular. In the future, the datasets with terabytes (10^2 bytes) will also become popular because of using high technology tools in collecting data. Then, to achieve effectively in extracting useful knowledge from such huge datasets, clustering algorithms must be efficient and scalable. In other words, the running time of a clustering algorithm must be predictable and acceptable in very large datasets. There are various methods to achieve requirements. In our thesis, the parallel techniques will be used to solve the problem.

- **Ability to deal with different types of attributes:** Besides the increasing in size of databases, the structure of the datasets is becoming much complex. It is popular that there are various attributes types in the same database. Each kind of attribute just suitable to certain similarity measures. Therefore, developing the similarity measure to deal with the database which has complex structure is still a challenge.
- **High dimensionality:** The size of database increases not only in number of objects but may also in number of attributes. So that, the dimensionality of the problem is high. A high dimensional data may lead to the increasing in size of search space for clustering algorithm. Besides, when the similarity measure depend on the number of the number of attributes, it is very much of time to spend for determine valuable clustering results. Approaches to this problem include method to reduce the effective dimensionality of the problem and the use of prior knowledge to identify irrelevant variables. In additions, to determine good clustering results, parallel techniques may be useful.
- **Discovery of clusters with arbitrary shape:** Many clustering algorithm use traditional similarity measures, such as Euclidean distance, Manhattan distance. Algorithms using this kind of distance measurement always tend to find spherical clusters with similar density and size. This limits the applying of the algorithms to different shapes of data
- **Minimal requirements for domain knowledge to determine input parameters:** Clustering algorithms expanded k-means paradigm and many others normally require some input parameters to process the task.

Determining acceptable these parameters needs prior domain knowledge. However, they are hard to estimate in some cases, especially for databases containing high-dimensional objects. Clustering accuracy may degrade drastically if a clustering algorithm is too sensitive to these input parameters. This not only burdens users, but also makes the quality of clustering difficult to control.

- **Able to deal with noise and outliers:** Outliers or erroneous data is a common problem in data analysis, especially, clustering. These sometimes are the reasons of missing accuracy results. A robust clustering algorithm should minimize or overcome the affect of this problem.
- **Able to deal with missing data:** Collecting huge database is a work which needed to be processed in long time and maybe in many places, etc. Therefore, it is normal if the database consists of missing data. In some databases, the number of missing values is noticeable. It means that, to work with this kind of database, the techniques to deal with missing values are necessary.
- **Insensitive to order of input records:** Clustering is the data mining task tries to finding natural groups of data objects from un-supervised database. Of course, it is easy to see that the problem is quite difficult. To get acceptable results, many algorithms require input parameters, which can be achieved only with experiences, and, therefore is impossible for some cases.

We will use these requirements to evaluate our clustering method in the conclusion of this thesis.

2.3.2 Database and Similarity Measure

One important and difficult problem when process clustering task is determining suitable similarity measure. The measure of similarity contributes much on quality of clustering results. However, the similarity between attribute values is depending much on data types. The following will discuss on some different types of data and their respective common similarity measures. [19]

2.3.2.1 Interval-scaled variables and the similarity measure

The section discusses interval-scaled variables and popular distance measures which are used to calculate the dissimilarity (or similarity) of data objects.

Interval-scaled variables are continuous measurements of a roughly linear scale. Example: weight, age, etc. Different measurement units of the same interval-scaled variable may make the different in processing of clustering analysis. Example, if the measurement unit changes from kilograms to pound then the range between data objects will also is changed and thus, effect on the resulting clustering structure. To avoid this problem, it is necessary to standardize the data.

Converting the original measurements to unitless variables is a well-known method. Given measurements for a variables *f*, we can convert it as following:

- Calculate the **mean absolute deviation**, s*f*

$$sf = \frac{1}{n}\left(\left|x_{1f} - m_f\right| + \ldots + \left|x_{nf} - m_f\right|\right), \tag{1}$$

where $x_{1f},\ldots,x_{nf}$ are *n* measurements of *f*, and m_f is the *mean* value of *f*, that is,

$$m_f = \frac{1}{n}\left(x_{1f} + \ldots + x_{nf}\right) \tag{2}$$

- Calculate the **standardized measurement**:

$$z_{if} = \frac{x_{if} - m_f}{s_f} \tag{3}$$

In the above two deviations, the mean absolute deviation is more robust to outliers than the standardized deviation.

In the research, we use standardized measurement to convert interval-scaled variables to unitless variables.

After convert the data, similarity between data objects described by interval-scaled variables is computed based on the distance between each pair of objects. The most popular distance measures are Euclidean distance and Manhattan distance.

2.3.2.2 Nominal Variables and the similarity measure

A nominal variable can have many states. For example, the day in a week is a nominal variable that may have, say, 7 states: *Monday, Tuesday, Wednesday, Thursday, Friday, Saturday,* and *Sunday.*

Let the number of states of a nominal variable be M. The states, then, can be denoted by numbers, letters such as $a_1, \ldots, a_M$.

The similarity between two objects *i* and *j* can be computed as:

$$d(i, j) = \frac{p-m}{p} \tag{4}$$

where *m* is the number of *matches* (i.e., the number of variables for which *i* and *j* are the same state), *p* is the number of total variables (or attributes).

2.3.2.3 Mixed data and the similarity measure

Because the databases are very huge and contain lot of information, therefore, they also are complex (i.e. having different kinds of attribute in the same database). It means that, the variables of data not only either interval-scaled variables or nominal variables but also includes both of them and furthermore, the other types of variable. This means that, the similarity measure will become more complex. Such as:

Let $Sim(o_i, o_j)$ be the similarity measure between o_i and o_j . $Sim(o_i, o_j)$ is defined as:

$$Sim(o_i, o_j) = \sum_{p=1}^{m} \delta^p (o_i, o_j) / m \tag{5}$$

where δ^p is defined as follows,

$$\delta^p (o_i^p, o_j^p) = \begin{cases} 1, \text{ if } o_i^p = o_j^p \text{ and } A_p \text{ is categorical} \\ 1, \text{ if } \left| o_i^p - o_j^p \right| < \mu \text{ and } A_p \text{ is numerical} \\ 0, \text{ otherwise.} \end{cases} \tag{6}$$

where µ is a threshold parameter defined by the user between [0, 1]. [34]

2.3.3 Clustering techniques

In this section, we will take a view on clustering techniques, especially, the techniques related with categorical and mixed data cause our research aim at propose clustering algorithms for categorical and mixed data.

In general, there are various clustering techniques. The goodness of each technique depends much on the motivation of the clustering task and also the kind of database, the dissimilarity measure.

Partitioning algorithms: Construct various partitions and then evaluate them by some criterion

Hierarchical algorithms: Create a hierarchical decomposition of the set of data (or objects) using some criterion

Density-based algorithms: based on connectivity and density functions

Model-based: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other.

Recent clustering methods for categorical and mixed data can be divided into two main branches. The first focuses on partitioning techniques and the second focuses on hierarchical techniques. Partitioning direction attempts to break a data set into *k* clusters such that the partition optimizes a given criterion [24], [33], [39], [6]. Centroid-based approaches, as typified by *k*-means [24] and ISODATA [2], try to assign points to clusters such that the mean square distance of points to the centroid of the assigned cluster is minimized. Centroid-based techniques are suitable only for data in metric spaces (e.g., Euclidean space) in which it is possible to compute a centroid of a given set of points. Medoid-based methods, as typified by PAM (Partitioning Around Medoids) [33] and CLARANS [39], work with similarity data, i.e., data in an arbitrary similarity space. These techniques try to find representative points (medoids) so as to minimize the sum of the distances of points from their closest medoid. Many partition based clustering algorithms tried to do with categorical and mixed data such as: Huang [21] proposed the k-modes algorithm based on the k-means paradigm to cluster categorical data by using modes instead of means for clusters. Huang and Ng [23], [45] also proposed a fuzzy k-modes algorithm by extending the fuzzy k-means. However, these two algorithms are unstable due to the non-uniqueness of the modes. That is, the clustering results depend on the selection of the modes. To overcome that, Ng and Wong [38], [43] tried to extend the k-modes algorithm with the Tabu search technique. Sun et al. [45], [27] proposed enhancement strategies for k-modes and fuzzy k-modes algorithms using an iterative initial-point refinement. Although these techniques can improve the quality of clusters, they are very time-consuming.

Hierarchical direction aims to produce a nested sequence of clusters, with a single all-inclusive cluster at the top and single point clusters at the bottom. Agglomerative hierarchical algorithms [24] start with all the data points as a separate cluster. Each step of the algorithm involves merging two clusters that are the most similar. After each merge, the total number of clusters decreases by one. These steps can be repeated until the desired number of clusters is obtained or the distance between two closest clusters is above a certain threshold distance. There are many different variations of agglomerative hierarchical algorithms [24]. These algorithms primarily differ in how they update the similarity between existing clusters and the merged clusters. In some methods [24], each cluster is represented by a centroid or medoid of the points contained in the cluster, and the similarity between two clusters is measured by the similarity between the centroids/medoids of the clusters. Like partitional techniques, such as K -means and K -medoids, these method also fail on clusters of arbitrary shapes and different sizes. Many hierarchical also tried to process with complex data, which contains categorical and mixed data. CACTUS [12] is an approach that entails the computation of summaries from the underlying data is sufficient to compute a set of “candidate” clusters which can then be validated to determine the actual set of clusters. The disadvantage of this algorithm is that running time of CACTUS is growing so faster when the number of dimensions grows. ROCK [17] is a hierarchical algorithm that uses a new concept of links—computed as the number of common neighbors between two objects—to measure the similarity/proximity between a pair of data points. ROCK can discover good quality clusters but its running time is still quite high.

Recently, Le and Ho [34] proposed a clustering method for categorical and mixed data, called the k-sets algorithm for categorical and mixed data. Unlike the other methods, this algorithm does not need to select the initial points (modes) of the clusters. Therefore, the results are stable. In addition, it was shown in [34] that the clustering accuracy is much better than those of previous clustering algorithms [21], [22], [23], [38], [45] when the performance was evaluated on the same databases collected from the UCI repository.

2.3.4 Parallelization in Clustering

The huge size of the available data sets and their high-dimensionality make large-scale data mining as well as data clustering applications computationally very demanding, to an extent that high-performance parallel computing is fast becoming an essential component of the solution. Moreover, the quality of the data mining results often depends directly on the amount of computing resource available. In fact, data mining applications are poised to become the dominant consumers of supercomputing in the near future. There is a necessity to develop effective parallel algorithms for various data mining techniques. However, designing such algorithms is challenging [26].

It is more useful to see that clustering techniques find interesting and previously unknown patterns in large scale data, embedded in a large multi-dimensional space and are applied to a wide variety of problems like customer segmentation based on similarity of buying interest [47], detection of clusters in geographic information systems [39], etc. Then, clustering algorithms need to efficiently scale up with the multi-dimensional of data sets and also the huge size of data. The aforementioned sequential clustering algorithms have a common problem that prevents them from running in a reasonable time when applied to a huge database on serial computers. To deal with huge databases, the people have tried to develop algorithms with small processing times. However, there still are clustering algorithms which can not possibly deal with a very large database in an acceptable time. Therefore it is necessary to develop parallel algorithms that can deal with very huge databases. Sanpawat Kantabutra et al. [30]and K. Stoffel et al. [43]proposed parallel clustering algorithms based on the k-means clustering algorithm. Their parallel k-means algorithms, however, are only to numeric databases only. Also, most other parallel clustering algorithms can deal with the databases consist of numeric attributes only [9], [27]

2.4 Summary

There are many techniques have been developed for the clustering task. However, it is worth noting that clustering with mixed data is still a challenge and become much more difficult when applying to a very large dataset. Therefore, there is a necessity of developing algorithms that can deal with very large and mixed databases.

The role of similarity measure is very important in a clustering algorithm, it influence the quality of the result so that the key point of developing a suitable clustering strategy for a database is strongly based on finding an appropriate similarity measure for all objects. In the next two chapters, we will discuss about HAC algorithms and parallel version of it to approach the solution.

Chapter 3. High Accuracy Clustering Algorithm for Categorical and Mixed Data

3.1 Introduction

There were many clustering algorithms, which expanded k-means paradigm, have been proposed for various kinds of databases, includes categorical and mixed data. However, the algorithms still have some drawbacks such as stopping at local minimum, depending on the initial selections of the means, modes, i.e. depending on the input order of data objects. One clustering algorithm for categorical and mixed data which can somehow overcome the problems was proposed by Le [34]. The algorithm chose k-largest sets from Non-Expandable Strongly Connected sets, which had been built by using Breath First Search algorithm. Therefore, it is not depending on the selecting points like means or modes. In addition, the remaining objects will be assigned to clusters by testing the minimum “distance” of the object with all clusters, consequently, the algorithm has no drawback in stopping at local minimum. However, the accuracy of the k-means algorithm is not always high. To improve the accuracy of clustering result, we propose a new clustering algorithm, which was improved from k-sets by using other methodology in re-partition step (step3) of k-sets.

In this chapter, first, the detail of the k-sets algorithm will be showed. And, then, we will discuss on our proposed improved algorithm. At the last of the chapter, we evaluate the accuracy of the proposed algorithm and also compare the results achieved from experimentally running the improved algorithm to the results obtained with previous algorithms on the same databases.

3.2 K-sets Clustering Algorithm for Categorical and Mixed Data

3.2.1 *Definitions*

Let D = $\{o_1, o_2, ..., o_n\}$ denote a set of n data objects; each object is described by m attributes $A_1, A_2, ..., A_m$, i.e., each o$_i$ is represented by a tuple, say $o_i = \{o_i^1, o_i^2, ..., o_i^m\}$, where $o_i^p \in dom(A_p)$, i = 1,…, n and p = 1,…, m.

3.2.1.1 *Similarity measure*

Let $Sim(o_i, o_j)$ be the similarity measure between o_i and o_j. $Sim(o_i, o_j)$ is defined as:

$$Sim(o_i, o_j) = \sum_{p=1}^{m} \delta^p(o_i, o_j)/m \qquad \textbf{(7)}$$

where δ^p is defined as follows,

$$\delta^p(o_i^p, o_j^p) = \begin{cases} 1, \text{ if } o_i^p = o_j^p \text{ and } A_p \text{ is categorical} \\ 1, \text{ if } \left|o_i^p - o_j^p\right| < \mu \text{ and } A_p \text{ is numerical} \\ 0, \text{ otherwise.} \end{cases} \qquad \textbf{(8)}$$

where µ is a threshold parameter defined by the user between [0, 1]. Since µ appears only when o_i^p and o_j^p are numerical data, µ can be ignored if the data objects are categorical data.

Obviously, Sim is symmetric for all pairs of data objects (o$_i$, o$_j$), we have Sim(o$_i$, o$_j$) = Sim (o$_j$, o$_i$).

3.2.1.2 *Neighbors*

Given a neighbor threshold θ, two objects o_i and o_j satisfy the neighbor relation if their similarity is greater than θ:

$$Neighbor(o_i, o_j) = \begin{cases} 1, & \text{if } Sim(o_i, o_j) \geq \theta \\ 0, & \text{otherwise} \end{cases} \qquad \textbf{(9)}$$

where θ is a threshold parameter defined by the user which changes uniformly from $\frac{1}{m}$ to 1 with the interval $\frac{1}{m}$.

3.2.1.3 Connection relation and strongly connected sets

From the neighbor relation, the connection relation is defined as follows. An object o_j is called connected to o_i if there is a sequence of objects: o_i = $o_{k1}, o_{k2}, ..., o_{kp}$ = o_j, where Neighbor(o_{kh}, o_{kh+1}) = 1 for h = 1, ..., p-1.

A set S of data objects is called a strongly connected set if for any two objects o_i, o_j $\in$ S, o_j is connected to o_i.

3.2.1.4 Non-Expandable Strongly Connected Sets (N-ESC sets)

A set of objects *CS* is called a *N-ESC set* if *CS* satisfies the following conditions:

(1) *CS* is not empty.

(2) *CS* is a strongly connected set.

(3) If o_j is connected to o_i and $o_i \in CS$ then $o_j \in CS$.

3.2.1.5 The Closest Core

Assume that we have *k* cores $CS_1,...,CS_k$ and an object o_i. The distance between o_i and a core CS_j is defined as

$$Dis(o_i, CS_j) = 1 - \max\{Sim(o_i, o_j) \mid o_j \in CS_j\} \qquad \textbf{(10)}$$

then, CS_j is called the *closest core* to o_i if

$$Dis(o_i, CS_j) = \min\{ Dis(o_i, CS_t) \mid t = 1, \ldots, k\} \qquad \textbf{(11)}$$

3.2.2 K-sets Clustering Algorithm

Figure 2 describes the the k-sets clustering algorithm proposed by Le and Ho [9].

```
Algorithm k-sets ($D, k, \theta, [\mu]$)
Input: $D = \{o_1, o_2, \ldots, o_n\}$, k: the number of clusters to be found, $\theta$: the neighbor threshold to
        determine whether two objects are neighbor or not, $\mu$: the threshold for numeric attributes.
    For each value of $\mu$, $\theta$, $k$ do:
        Step1. /*Find non-expandable strongly connected sets (N-ESC sets*/
                $l = 0$.
                For each $oi \in D$ if $oi \notin \cup S_t$ ($t = 1, ..., l$) then $l = l+1$, $S_l = \{o_i\}$.
                Build the N-ESC set $S_l$ as defined.
        Step2. /*Choose $k$ cores for $k$ clusters being found*/
                Choose $k$ largest sets from $l$ N-ESC sets $S_1, S_2, \ldots, S_l$ to be $k$ cores $CS_1, \ldots, CS_k$ for
                $k$ clusters being found.
        Step3. /*Find clusters for data objects*/
                For each $o_i \in D$,
                    Find the closest core $CS_j$ for $oi$.
                    Assign $o_i$ to cluster contains $CS_j$.
Output: $k$ clusters.
```

Figure 2: Scheme of the k-sets algorithm

As shown in Figure 2, the algorithm consists of three steps. The first step is to find all the non-expandable strongly connected sets (N-ESC sets) of *D* with threshold θ. Next, the *k* largest sets are chosen among *l* N-ESC sets of $S_1, \ldots, S_l$ as *k* cores of *k* clusters being found. Finally, each data object o_j is assigned to the cluster whose core is the closest set to the object. If there is more than one cluster satisfying this condition, the cluster with largest core will be selected in step 3. The algorithm terminates when all objects are assigned to their own classes.

3.2.3 Computational Complexity of the k-sets Algorithm

At step 1, a breadth first search algorithm is used to find N-ESC sets of n objects.. The computational complexity of this process is $O(n^2)$. Step 2 takes $O(l \log l)$ to sort the l N-ESC sets found above to find k core of k clusters being found. In step 3, $|CS_1|$ +

$|CS_2|$ + …+ $|CS_k|$ distances are considered for each object. The computational complexity of this process is $O(n)$. Then, for n objects, the complexity is $O(n^2)$.

Therefore, for each value of μ, θ, k the total complexity is $O(n^2)$.

Take the number of possible values of μ, θ, k into account, the complexity of the algorithm is $O(|\mathsf{M}| * |\mathsf{K}| * m * n^2)$. In that, M is set of possible values of μ, K is set of possible values of k, and m represents the maximum possible values of θ. If the data consists of only categorical attributes, the value of μ can be ignored. In that case, the complexity will be $O(|\mathsf{K}| * m * n^2)$.

3.3 High Accuracy Clustering Algorithm for Categorical and Mixed Data

In this section, we propose an improvement of the k-sets clustering algorithm. We also compare the results achieved from experimentally running the improved algorithm to the results obtained with previous algorithms as well as the performance of the parallel algorithm.

3.3.1 Improvement Approach based on a new Grouping Approach

Once the first step of the original k-sets algorithm have done, the database D = $\{o_1, o_2, ..., o_n\}$ is separated to set of *l* Non-Expandable Strongly Connected (N-ESC) sets $S_1, S_2, ..., S_l$. Note that, each N-ESC set consists of all data objects which connected with each other by *connection relation* as showed in 3.2.1.3. Consequently, objects in the same N-ESC set can be considered as more similar than objects in different N-ESC sets.

Next, after choosing *k* largest N-ESC sets to be *k cores* of *k* clusters, there are objects in the remaining *l-k* N-ESC sets need to be assigned to their own clusters. To find suitable cluster for remaining objects, the k-sets do as following:

Let's $o_i,\ o_j,...,$ are objects in the N-ESC set *h*, then the k-sets find the closest core CS_{hi} , CS_{hj}, …, for $o_i,\ o_j,...,$, respectively. Then, assign $o_i,\ o_j,...,$ to their respective clusters which contain their closest cores. This re-partitioning method may lead to the case that objects $o_i,\ o_j$ are the same N-ESC set *h* but are assigned to different clusters. However, the objects in the same N-ESC set are more similar than the objects in different N-ESC sets as noticed above.

In additions, the re-partitioning method assigns an object to cluster depend on the minimum distance to certain element in that cluster. If the minimum distance is the distance between the object and an outlier of the cluster, then the object will be incorrectly assigned to the cluster cause the outlier. Therefore, using this re-partitioning method may get low accuracy results cause the outlier.

Aim at achieve higher accuracy in clustering results we proposed another method to do this re-partitioning step. In stead of finding closest core for each object o_i, o_j in the N-ESC *h* set, we find one *closest core* for the N-ESC set *h*. The *closest core* of a set is defined as:

The closest core of a set:

Assume that we have k cores $CS_1,...,CS_k$ and a set S_i. The distance between S_i and a core CS_j is defined as

$$Dis(S_i,CS_j)=1-\frac{\sum_{\forall o_i\in S_i}\max\{Sim(o_i,o_j)\mid o_j\in CS_j\}}{|S_i|} \tag{12}$$

and CS_j is called the *closest core* of S_i if

$$\text{Dis}(\text{S}_i, \text{CS}_j) = \min \{ \text{Dis} (\text{S}_i, \text{CS}_t) \mid t = 1, \dots, k\} \tag{13}$$

According to our proposed approach, after re-partitioning, the objects in the same N-ESC set are still in the same cluster. Consequently, the approach hopefully to give us higher accurate clustering results in certain databases and avoid the mis-clustering cause by outlier.

The Figure 3 shows an example of our approach to make clearly in understanding.

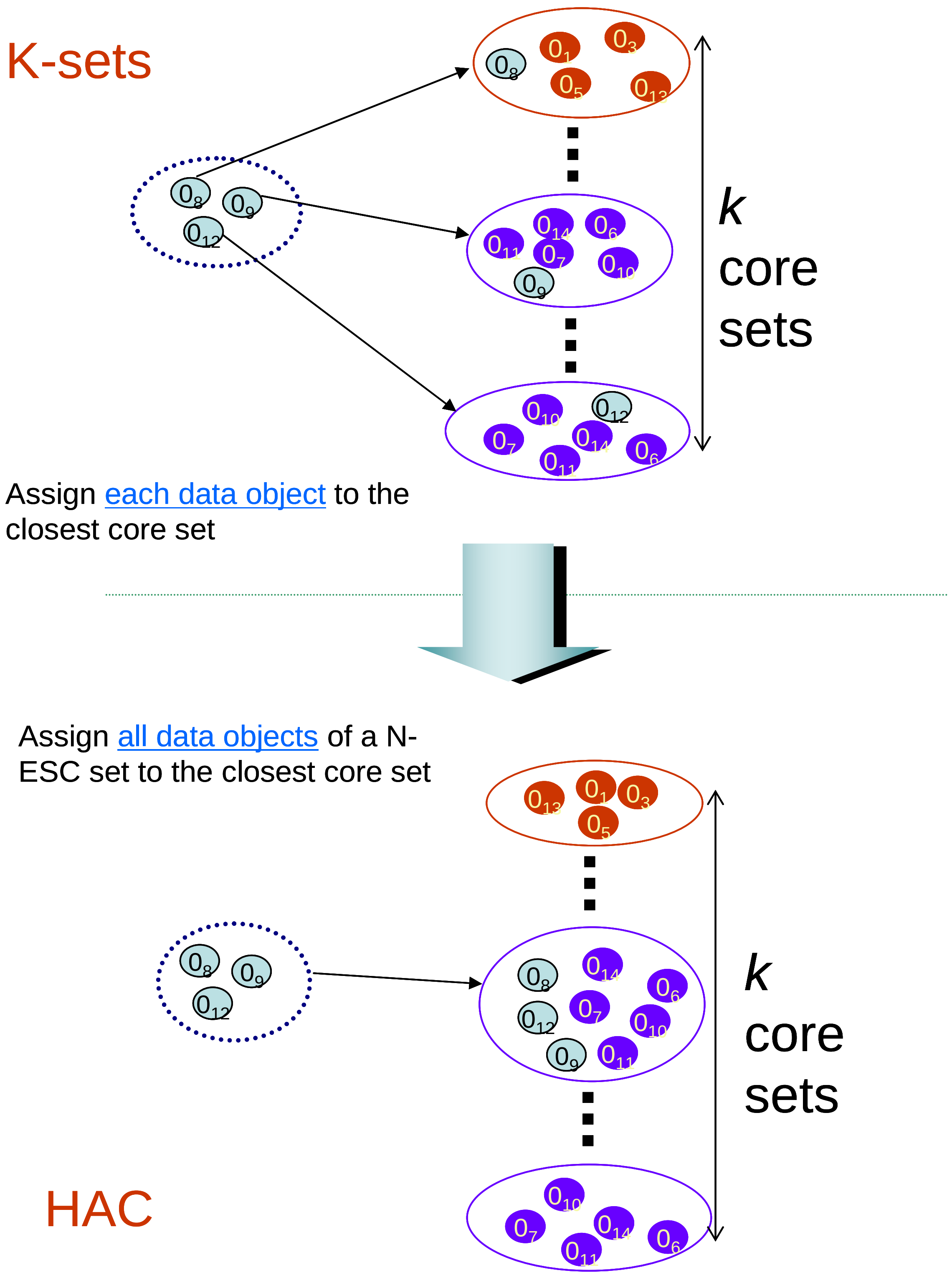

Figure 3: Improvement Approach Based on a new Grouping Approach.

3.3.2 High Accuracy Clustering Algorithm for Categorical and Mixed Data based on a New Grouping Approach

All the definitions given in 3.2.1 and 3.3.1 are used for presenting the High Accuracy Clustering Algorithm for Categorical and Mixed Data based on a new Grouping Approach (HAC algorithm).

Figure 4 shows the HAC algorithm.

```
HAC algorithm (D, k, θ, [μ])
Input: $D = \{o_1, o_2, \ldots, o_n\}$, k: the number of clusters to be found, θ: the neighbor threshold to
        determine whether two objects are neighbor or not, μ: the threshold for numeric attributes.
   For each value of μ, θ, k do:
        Step1. /*Find non-expandable strongly connected sets (N-ESC sets)*/
               $l = 0$.
               For each $oi \in D$ if $oi \notin \cup S_t$ $(t = 1, ..., l)$ then $l = l+1$, $S_l = \{o_i\}$.
               Build the N-ESC set $S_l$ as defined above
        Step2. /*Choose k core for k cluster being found*/
               Choose k largest sets from l N-ESC $S_1, S_2, \ldots, S_l$ to be k core $CS_1, \ldots, CS_k$ for
               k clusters being found.
        Step3. /*Find clusters for remain sets*/
               /*This step is different from the original k-sets clustering algorithm*/
               For each $S_i \notin \{CS_1, \ldots, CS_k\}$
                   Find the closest core $CS_j$ for $S_i$
                    Assign all objects of $S_i$ to cluster contains $CS_j$.
Output: k clusters.
```

Figure 4: Scheme of the HAC algorithm

3.3.3 Computational complexity of the HAC algorithm

The difference between the k-sets clustering algorithm and the HAC algorithm is finding clusters i.e. find *closest core*) for the remaining sets. To find the *closest core* for N-ESC set S_i, it is necessary to consider $| S_i| * (|CS_1| + |CS_2| + \ldots + |CS_k|)$ distances. Then, to find the closets cores for $(l\text{-}k)$ N-ESC sets $S_{i1}, S_{i2}, \ldots, S_{i(l-k)}$, we must consider $(| S_{i1} | + | S_{i2} | + \ldots + | S_{i(l-k)} |) * (| CS_1 | + | CS_2 | + \ldots + | CS_k |)$ distances. The computational complexity of this phase is $O(n^2)$.

Consequently, the computational complexity of the HAC is $O(|\mathsf{M}| * |\mathsf{K}| * m * n^2)$ in case of mixed data or $O(|\mathsf{K}| * m * n^2)$ when the data consists of only categorical attributes. In that, M is set of possible values of μ, K is set of possible values of k, and m represents the maximum possible values of θ.

3.4 Performance Evaluation

In this section, the performance of the HAC algorithm will be showed to evaluate the clustering result of the algorithm. The experiments are done on the same databases used by related works, for a sound comparison.

3.4.1 Accuracy Calculation

Accuracy is calculated in the same way as in the papers whose results are compared with the results here. This accuracy measure is defined as follows:

$$r = \frac{\sum_{l=1}^{k} r_l}{n}, \qquad \textbf{(14)}$$

where r_l is the maximum number of data objects of cluster l belonging to the same original classes in the test data (correct answer) and n is the number of data objects in the database.

3.4.2 Databases

All three databases used in the experiments were obtained from the UCI machine learning repository [4]:

Database	Number of object	Number of numeric attribute	Number of categorical attribute	Number of total attribute

Soybean	47	0	35	35
Credit1	666	6	9	15

Table 2. Databases to be used for evaluation the accuracy of k-sets algorithm with the others.

3.4.3 Results

Table 3 shows the accuracy of clustering algorithms on the soybean disease data set. On soybean disease data set, our proposed HAC algorithm achieves the best accuracy at k=1, θ=0.714, and μ=0. The accuracy of our algorithm is the highest among the six algorithms shown in **Table 3**. (The results of others is taken from [34])

Algorithm	Accuracy
HAC algorithm	1.00
K-sets [34]	1.00
Fuzzy k-modes [23]	0.79
Tabu search based k-modes [38]	0.99
K-modes [21]	0.89
K-modes with refinement initialization [45]	0.98

Table 3. Accuracy of clustering algorithm on the soybean disease database (the accuracy of the other algorithms are obtained from [34]).

Table 4 shows the accuracy of clustering algorithms on the credit 1 database. For this database, the proposed HAC algorithm achieves the best accuracy at k=2, θ=0.93, μ=0.3. The proposed algorithm's accuracy is the highest compared to other methods, as shown in **Table 4**. (The results of the others is taken from [34])

Algorithm	Accuracy
HAC algorithm	0.8378
K-sets [34]	0.8288
K-prototypes [22]	0.77
Tabu search based k-prototypes [38]	0.80

Table 4. Accuracy of clustering algorithms on the credit 1 database (the accuracy of the other algorithms is obtained from [34]).

Running with two sub-databases: 500 and 1000 objects of the Connect-4 database, the proposed algorithm achieve the best accuracy at k=8, θ=0.976, μ=0 and k=98, θ=0.976, μ=0, respectively. The accuracy of the HAC algorithm in these databases is also higher than the accuracy of the k-sets algorithm. We can see these results in **Table 5** as follow:

Algorithm	Accuracy (with 500 objects)	Accuracy (with 1000 objects)
HAC algorithm	0.73	0.815
K-sets algorithm	0.718	0.805

Table 5. Accuracy of original k-sets and improved k-sets algorithm on the various sub-databases of connect-4-database

3.5 Summary

From these experimental results, we can see that the original k-sets clustering algorithm and the HAC algorithm can find clusters that are more similar to the original clusters than the other partitioning algorithms. The presentation of each cluster by a set of Non-Expandable Strongly Connected sets in the original k-sets algorithm and the HAC algorithm allow the discovery of more complex clusters in real-life databases.

However, the HAC algorithm can achieve better results than the original one. Therefore, we can conclude that the method to re-partition objects to core sets of our proposed HAC algorithm is reasonable.

In addition, the k-sets and our proposed HAC algorithm using Breath First Search algorithm and the *Connection Relation* to build *l* Non-Expandable Strongly Connected sets, then select *k* cores for final *k* clusters from such *l* N-ESC sets. Therefore, these algorithms do not depend on the input data. Furthermore, the *k* cores are chosen from all *l* N-ESC sets, which contain all data objects of the database. Consequently, the algorithms can get optimization in total, not only in local as the previous algorithms.

Chapter 4. Parallel HAC Algorithms

4.1 Introduction

Partitioning a set of large heterogeneous datasets into smaller homogeneous subsets that can be easily managed, separately modeled and analyzed is a fundamental operation in many fields, especially, data mining. Clustering is a popular approach used to implement this operation. Growing with developing of high data-collection techniques, the data is recently very huge and seem to be much huger in business, etc. Therefore, clustering task usually has to deal with large databases, which is impractical to process by using even scalable clustering algorithms. In addition, the technology of high performance computer is well developed. This enable the human can do task in acceptable time by parallelize sub-tasks. Consequently, one reasonable approach helps clustering algorithm to be able to deal with very huge databases is applying parallel techniques. Until now, there are some parallel clustering algorithms were proposed [43], [27], [9], [30]. However, most of the parallel clustering algorithms are developed from the serial clustering algorithms which are able to numeric data only. This limits their applicable to processing with categorical data or furthermore, mixed data.

Our proposed HAC algorithm is showed to be able to achieve high accuracy clustering result. This following by using HAC algorithm make possible to find good description of the database, the aim of clustering task. Unfortunately, despite running on a very powerful computer, the execution time of the proposed HAC algorithm is still very high for large databases.

The above facts encourage us to propose parallel clustering algorithm based on our proposed HAC algorithm. In this chapter, two parallel algorithms based on the proposed HAC algorithm will be showed and discussed in detail. The first parallel algorithm named *Single-Level Parallel HAC algorithm,* will parallelize the HAC algorithm at trials of the input threshold θ . The second one has the name with *Multi-Level Parallel HAC algorithm.* In this parallel algorithm, not only the trials of θ but also the trials of

μ are parallelized to reduce processing time when deal with huge databases by using large enough number of processors.

4.2 Single-Level Parallel HAC Algorithm

4.2.1 The signification of applying parallelization for HAC algorithm

In spite of the significant improvement of the C version performance, the execution time of the HAC algorithm with large datasets is still high. Figure 5 shows the measured execution time of the HAC algorithm with four sub-datasets of connect-4 database. These execution times was measured by implementation of HAC clustering algorithm on one processor of Cray T3E.

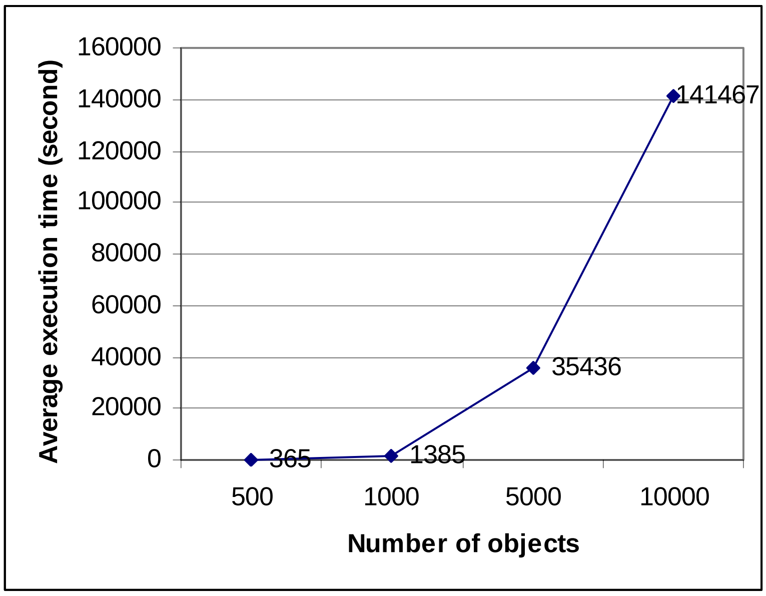

Figure 5: Execution time of the HAC algorithm with various sub-databases of Connect-4 database

In Figure 5, the execution time of HAC algorithm increases rapidly in proportional of $(n)^2$. Therefore, the HAC algorithm should be parallelized in order to deal with huge datasets.

4.2.2 Discussion of HAC Algorithm to get Single-Level Parallel Approach

Let's analyze the detail of execution time of HAC algorithm with connect-4 dataset, which consist of 5000 objects and each object is described by 42 attributes.

For each value of input parameters:μ,θ, k, we have:

The execution time of step 1, step 2 and step 3 is 0.176 seconds, 0.0097 seconds and 84.19 seconds, respectively.

To achieve high degree of accuracy, it is necessary to run the k-sets algorithm with all values of μ (0.1, …, 1, interval of 0.1) and θ (1/m, …, m/m, interval of 1/m) and also various values of k. In this case, because the connect-4 dataset is categorical data, therefore, as we analyzed above, the parameter μ can be ignored. Finally, with 42 different values of θ and 10 different value of k, the total execution time of HAC clustering algorithm is 42*10*(0.176+0.0097+84.19)≈35436 seconds.

It is easy to see that the total execution time of whole HAC algorithm is significantly larger than the total execution time of 3 main steps of the algorithm. Therefore, the HAC algorithm should be parallelized on the trials of input parameters.

4.2.3 The Single-Level Parallel Approach

According to the analysis in the previous section, the trials for μ, θ and k is large and consequently, take a considerable processing time. Therefore, we consider to apply parallel technique to these trials. We also analyzed in the section 3.2.1.1 that the input parameter μ can be ignored when the dataset consist of only categorical data as discussed above. In addition, we can somehow determine value of k in small interval, then, the trials for k will not be so much. Consequently, we do not apply parallel technique for the trial with μ and k but parallelizing the trial with θ by equally separate values of θ over processors.

Each processor will execute the same code (the code of step 1, step 2 and step 3 of the improved algorithm) with the same trials with μ, and k but different trials with θ. The trials with θ on each processor are determined as the following:

For each loop with each value of μ and k, there are at most (m+1) trials with θ as discussed in the section 3.2.1.2. Assume we have P processors. (m+1) trials with θ will be equally separated to P processors. In general, once divide the number of trials with θ (m+1) to the number of processors P, the results contains both the quotient $\left[\frac{m+1}{p}\right]$ and the remaining ((m+1) mod P). Therefore, P processors will be separated into two groups. The first group has (P – ((m+1) mod P) processors and the second has ((m+1) mod P) processors. Processors of both two groups will executes three steps of the improved algorithm with the same number of trials with μ, k but $\left[\frac{m+1}{p}\right]$ trials with θ on each processor of the first group and $\left[\frac{m+1}{p}\right]$+1 trials with θ on each processor of the second group. This approach hopefully keeps balance in total processing time among all processors because tasks of two processors are just different with only one loop at most.

The following illustration shows the approach clearly.

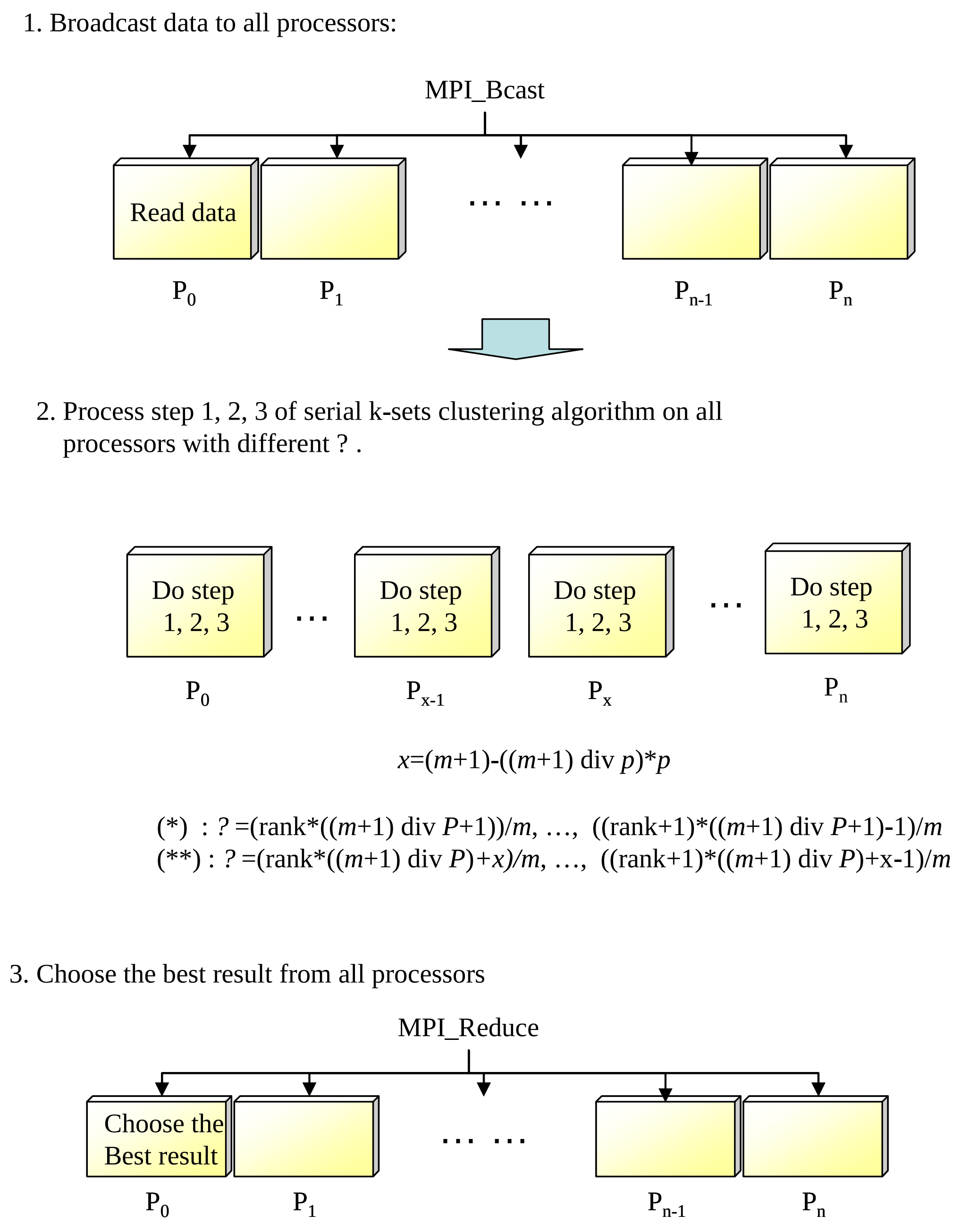
1. Broadcast data to all processors:
MPI_Bcast
Read data
… …
P0
P1
Pn-1
Pn
2. Process step 1, 2, 3 of serial k-sets clustering algorithm on all processors with different ? .
Do step 1, 2, 3
…
P0
Px-1
Px
Pn
x=(m+1)-((m+1) div p)*p
(*) : ? =(rank*((m+1) div P+1))/m, …, ((rank+1)*((m+1) div P+1)-1)/m
(**) : ? =(rank*((m+1) div P)+x)/m, …, ((rank+1)*((m+1) div P)+x-1)/m
3. Choose the best result from all processors
MPI_Reduce
Choose the Best result
… …
P0
P1
Pn-1
Pn

Figure 6: Illustration of the Single-Level Parallel HAC algorithm

4.2.4 Scheme of the Single-Level Parallel HAC algorithm

Figure 6 shows the proposed parallel algorithm. First, to save time, only one processor P_0 is eamployed to read data and then the data are broadcasted to other PEs. Second, all processors execute k-sets. The difference on processors is the value of θ as shown in the Figure 6. Finally, P_0 use MPI_Reduce command to choose the highest performance result from all processors.

```
Single-Level Parallel HAC Algorithm ($D$, $k$, $\theta$, [$\mu$])
Input: $D = \{o_1, o_2, \ldots, o_n\}$, k: the number of clusters to be found, $\theta$: the neighbor threshold to
        determine whether two objects are neighbor or not, $\mu$: the threshold for numeric attributes.
    i. Data is read by one processor and is broadcasted to all other processors.
    ii. On each processor do
        For each value of $\mu$
            For each value of $\theta$ (value of $\theta$ is different on each processor)
                For each value of $k$ do
                    Step1. /*Find non-expandable strongly connected sets (N-ESC sets)*/
                        $l = 0$.
                        For each $oi \in D$ if $oi \not\subseteq \cup S_t$ ($t = 1, \ldots, l$) then $l = l+1$, $S_l = \{o_i\}$.
                        Build the N-ESC set $S_l$ as defined above
                    Step2. /*Choose k core sets*/
                        Choose $k$ largest sets from $l$ N-ESC sets $S_1, S_2, \ldots, S_l$ to be $k$ cores $CS_1, \ldots, CS_k$
                        for $k$ clusters.
                    Step3. /*Find clusters for remain N-ESC sets*/
                        For each $o_i \in D$,
                            Find the closest core $CS_j$ for $oi$.
                            Assign $o_i$ to cluster contains $CS_j$.
        Keep the best of the recent result and the previous result.
    iii. Choose the best result from results on all processors.
Output: $k$ clusters.
```

Figure 7: The Single-Level Parallel HAC Algorithm

4.2.5 *Computational Complexity of the Single-Level Parallel HAC Algorithm*

To load the whole dataset in phase (i), the time required is O(n). In the phase (ii), on each processor, O(n^2) is needed to execute step 1, step 2 and step 3, as discussed in section 3.3.3. Therefore, the computational complexity for phase 2 is $O(|\mathrm{M}| * |\theta_P| * |\mathrm{K}| * n^2)$. In that, M is set of possible values of μ, K is set of possible values of k, and $\theta_P \approx \frac{m}{P}$ is the number of values of θ on each processor. However, μ can be ignored when the data is categorical data, therefore, the time complexity for this phase in the case is $O\left(\frac{m}{P}|\mathrm{K}|n^2\right)$; m is the number of attributes, n is the number of objects and P is the number of processors. The time complexity of the phase (3) is O(P).

Finally, the computational complexity of the parallel algorithm is $O\left(|\mathrm{M}|\frac{m}{P}|\mathrm{K}|n^2\right)$ when the data is mixed data, or $O\left(\frac{m}{P}|\mathrm{K}|n^2\right)$ for categorical data. This means that the proposed parallel clustering algorithm is scalable in theory.

4.3 Multi-Level Parallel HAC Algorithm

4.3.1 The extension approach

The Single-Level Parallel approach has been proposed for HAC algorithm to reduce execution time. The algorithm considers only trial of θ. However, in case of mixed data, the processing time spend for trial of μ also very large as discussed previous. Therefore, we would like to apply multi-level parallel techniques for the HAC algorithm. It seems to be effective and scalable in case of mixed data. We have the approach of the parallel algorithm as follow:

Each processor will execute the same code (the code of step 1, step 2 and step 3 of the improved algorithm) with the same trials for k but different trials for θ and the same or different trials for μ (depends on the number of processor and number of possible value of θ: totaltheta). The detail of parallelization approach is described as the following:

Let's called the number of possible value of θ is totaltheta (it is normally equal m, the number of attributes), and the number of possible value of μ is totalmiu. In addition, assume that there are P processors, thetaprocs is the number of groups of processors, with processors in the same group has the same values of θ and processors in different groups has different values of θ; miuprocs is the number of processors in one group of thetaprocs groups, processors in the same one group of sets of thetaprocs groups has different value of μ, and processors with the same orders in their group of thetaprocs groups has the same value of μ.

Let's difprocs=P/totaltheta, then, we have:

+ If the number of processor P is smaller than or equal to the number of totaltheta (difprocs≤1) then thetaprocs=P; thetamiu=1.

+ If the number of processors P greater than the number of possible value of θ : totaltheta, difprocs times which is greater than 1 and smaller than totalmiu, then we have two cases:

a. if P%totaltheta <= difprocs then the trials with θ will be equally separated to thetaprocs=totaltheta groups of processors, each group contains miuprocs=difprocs processors. Processors in the same group has the same value of θ. Then, on miuprocs processors of each group, the trials with μ will be equally separated.

b. if P%totaltheta > difprocs then the trials with θ will be equally separated to thetaprocs=

$\frac{P}{difprocs+1}$ groups of processors, each group contains miuprocs=difprocs+1 processors. Processors in the same group has the same value of θ. Then, on miuprocs processors of each group, the trials with μ will be equally separated.

+ If the number of processors P greater than the number of possible value of θ difprocs times which is greater than or equal to totalmiu, then, the trials with θ will be equally separated to thetaprocs=totaltheta groups of processors, each group contains miuprocs=totalmiu processors. Processors in the same group has the same value of θ. Then, on miuprocs processors of each group, the trials with μ will be equally separated. This partition in the condition also prevent sending data to non-used processors in case the number of processors greater than multiplication of total of possible value of θ and μ.

In general, the value of θ (labeled by theta) and μ (labeled by miu) on each processor is calculated by the following formulas:

$$theta = starttheta + \frac{myid}{miuprocs} trialtheta + i * thetaprocs * trialtheta \ , \qquad \textbf{(15)}$$

$$miu = startmiu + (myid\%miuprocs) * trialmiu + i * miuprocs * trialmiu,, \quad \textbf{(16)}$$

in which, starttheta, startmiu are the first possible value of θ: $\frac{1}{m}$, μ, respectively; myid is rank number of processor, trialtheta, trialmiu are the different between two values of θ , μ which adjoined each other; i is interger: i=0, 1, …, $i_{maxtheta}$ (i_{maxmiu}) so that theta, miu<=1.

Note that, the parameters: startmiu, starttheta, trialmiu, trialtheta, totaltheta, totalmiu can be input by the users. When the data is categorical data only, the value miuprocs equal 1 for all cases.

This approach hopefully balances total processing time among all the processors because tasks of two processors are different by at most only one loop. In addition, the proposed approach is hoped to increase much the processing time. Then, it can be used to deal with very large databases.

4.3.2 The Multi-Level Parallel HAC Algorithm

Error! Reference source not found.8 shows the proposed parallel algorithm. First, to save time, only one processor P_0 is employed to read data and the data are broadcast to other processors. Second, all processors execute k-sets. The difference among processors is the value of θ and may also value of μ as discussed in section 4.1. Finally, P_0 uses the MPI_Reduce command to choose the best result from all processors.

```
Multi-Level Parallel HAC Algorithm ($D, k, \theta$, [$\mu$])
Input: $D = \{o_1, o_2, \ldots, o_n\}$, k: the number of clusters to be found, $\theta$: the neighbor threshold to
        determine whether two objects are neighbor or not, $\mu$: the threshold for numeric attributes.
    i. Data is read by one processor and is broadcasted to all other processors.
    ii. On each processor do
          For each value of $\theta$ (value of $\theta$ is different on each processor and be calculated as in the formula 8
            For each value of $\mu$ (value of $\mu$ is different on each processor and be calculated as in the formula
                9)
              For each value of $k$ do
                  Step1. /*Find non-expandable strongly connected sets (N-ESC sets)*/
                    $l = 0$.
                    For each $oi \in D$ if $oi \notin \cup S_t$ ($t = 1, \ldots, l$) then $l = l+1$, $S_l = \{o_i\}$.
                    Build the N-ESC set $S_l$ as defined above
                  Step2. /*Choose $k$ cores for $k$ cluster being found*/
                    Choose $k$ largest sets from $l$ N-ESC sets $S_1, S_2, \ldots, S_l$ to be $k$ cores $CS_1, \ldots, CS_k$ for
                    $k$ clusters being found.
                  Step3. /*Find clusters for remain N-ESC sets*/
                    For each $S_i \notin \{CS_1, \ldots, CS_k\}$
                        Find the closest core $CS_j$ for $S_i$
                        Assign all objects of $S_i$ to cluster contains $CS_j$.
          Keep the best of the recent result and the previous result.
    iii. Choose the best result from results on all processors.
Output: $k$ clusters.
```

Figure 8: The Multi-Level Parallel HAC Algorithm

4.3.3 Computational Complexity

To load the whole dataset in phase (i), the time required is O(n). In the phase (ii), on each processor, O(n^2) is needed to execute step 1, step 2 and step 3, as discussed in section 3.3.3. Therefore, the computational complexity for phase 2 is $O(|\mathrm{M}_P|*|\Theta_P|*|\mathrm{K}|*n^2$, where Θ_P, M_P is the number of values of θ, μ, respectively, on each processor. However, μ can be ignored when the data is categorical data, therefore, the time complexity for this phase in the case is $O(|\Theta_P|*|\mathrm{K}|*n^2)$; n is the number of. The time complexity of the phase (3) is O(P).

Finally, the computational complexity of the parallel algorithm is $O(|\mathrm{M}_P|*|\Theta_P|*|\mathrm{K}|*n^2$ when the data is mixed data, or $O(|\Theta_P|*|\mathrm{K}|*n^2)$ for categorical data.

4.4 Summary

According to the necessity of doing clustering task with huge databases, it is expected to propose scalable clustering algorithms to process with huge data in acceptable time. However, the serial algorithm can not satisfy such requirement. One approach to solve the problem is using parallel techniques. In this section, we have proposed two parallel algorithms for our HAC algorithm. The analysis of parallel approaches and their computational complexity shows that they are effective and scalable parallel algorithms.

In the following chapter, we will show their performance and the analysis which prove that the proposed parallel algorithms are good enough.

Chapter 5. Performance Evaluation

5.1 Introduction

To estimate the effect of parallel techniques on the parallel algorithms of HAC, implementation of the algorithms were done on high perform computing with various data sets. We measured the average execution time, the speedup, efficiency of the algorithm. In addition, we show loosing time for parallel environment when running the algorithms on parallel computing. This loosing time somehow help to explain the results of achieved speedup and efficiency.

5.2 Database

The connect -4 opening database contains all legal 8-ply positions in the game of connect-4 in which neither player has won yet, and in which the next move is not forced.

KDD Cup 1999 is the database used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining. The competition task was to build a network intrusion detector, a predictive model capable of distinguishing between ``bad" connections, called intrusions or attacks, and ``good" normal connections. This database contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment.

Database	Number of object	Number of numeric attribute	Number of categorical attribute	Number of total attribute
Soybean	47	0	35	35
Credit1	666	6	9	15
Connect-4-1000	1000	0	42	42
Connect-4-5000	5000	0	42	42
Connect-4-10000	10000	0	42	42
Connect-4-50000	50000	0	42	42
KDD Cup 1000	1000	34	7	41
KDD Cup 5000	5000	34	7	41
KDD Cup 10000	10000	34	7	41

Table 6. Databases to be used to evaluate the performance of the parallel k-sets algorithm.

5.3 Implementation Environment

The proposed parallel algorithm is implemented on Cray T3E 1200E. Cray T3E consists of 128 nodes. Each node has 512 MB memory and a 600MHz Alpha 21164 CPU. We use Message Passing Interface (MPI) for communication between processors.

5.4 Parallel performance

5.4.1 Execution time

Execution time is the time that elapses while the program is executed. In the experiments, the parallel programs are executed 5 times to get the average execution times. Figure 9 shows the average execution times of the parallel HAC algorithms on difference numbers of processors with various datasets. To understand the graph easy, please note that: “Multi-Level Parallel – KDD Cup 1999 – 5000” this is the result of the Multi-Level Parallel HAC algorithm running with the subset 5000 data objects of the KDD Cup 1999 database. And it is similarly to the other notice in the figures.

It is easy to see that the total execution time decreases as the number of processors increases. In particular, the time is significantly decreases in case of largest datasets. We can conclude that the more increase of size of datasets and larger number of processors, the more decrease of execution time.

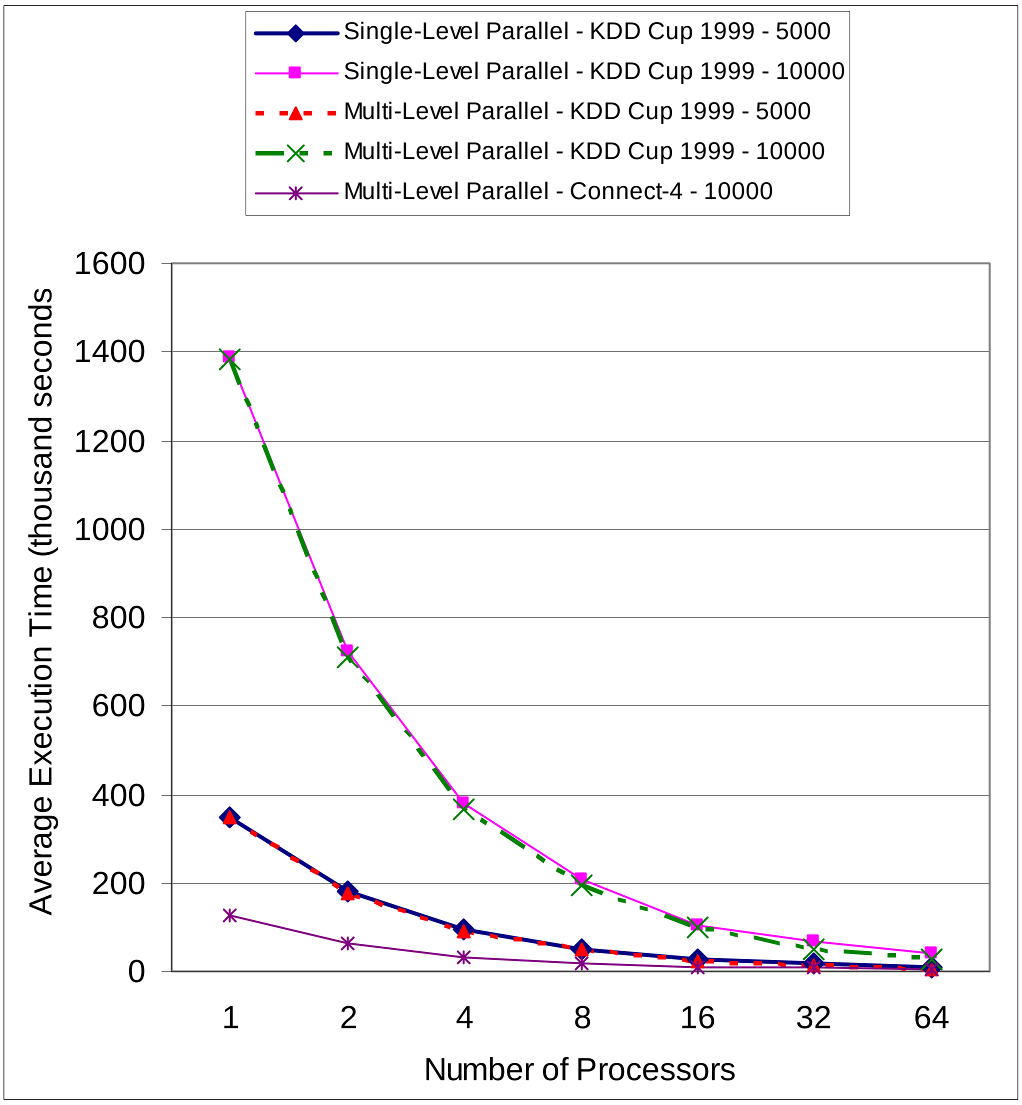

Figure 9: Average execution times of parallel HAC algorithms on different numbers of processors with various databases.

Figure 10 shows more clearly about the average execution time of the two parallel algorithms. We can see from the figure that, the average execution time of both two parallel algorithms is decreased much when the number of processors changes from 1 to 41. However, the average execution time of the Single-Level Parallel one do not change after the number of processors reach to 41. While the average execution time of the Multi-Level Parallel one still reduces significantly. The reason is that the Single-Level Parallel one parallelize the HAC algorithm at the threshold θ only, then, the maximum number of processors the algorithm can use is maximum number of trial of θ (this case is 41, the number of attributes in the database). Therefore, the performance of the Multi-Level Parallel one is much better than the performance of the Single-Level Parallel one when dealing with mixed data on large enough number of processors.

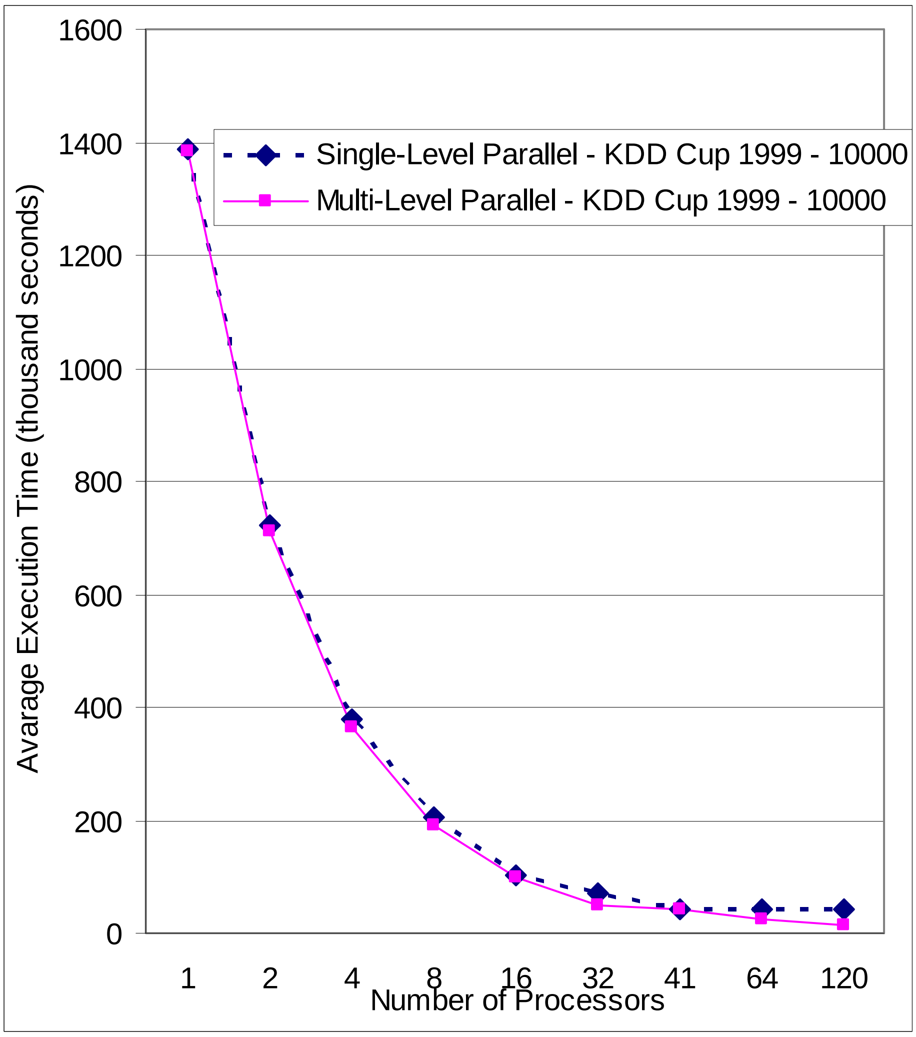

Figure 10: Average execution times of parallel HAC algorithms on different numbers of processors but the same database.

5.4.2 Speed up

Speedup is the ratio between running time of the best available sequential and running time of the parallel algorithm:

$$\text{Speedup} = \frac{\text{Running time of the best available sequential algorithm}}{\text{Running time of the paralell algorithm}} \qquad \textbf{(17)}$$

Figure 11 shows the speed up of the proposed parallel algorithms with the sub-dataset of KDD Cup 1999 of 10000objects. We can recognize that the parallel algorithm realizes high speed up on a greater number of parallel processors with this database. Similarly to the average execution time, the speed up of Single-Level Parallel algorithm is not increase from when the number of processors reach to 41 but the speed up of the Multi-Level Parallel algorithm is rapidly increase. It become similar to linear speedup and achieve 90 times faster than the serial HAC algorithm on 120 processors in this experiment.

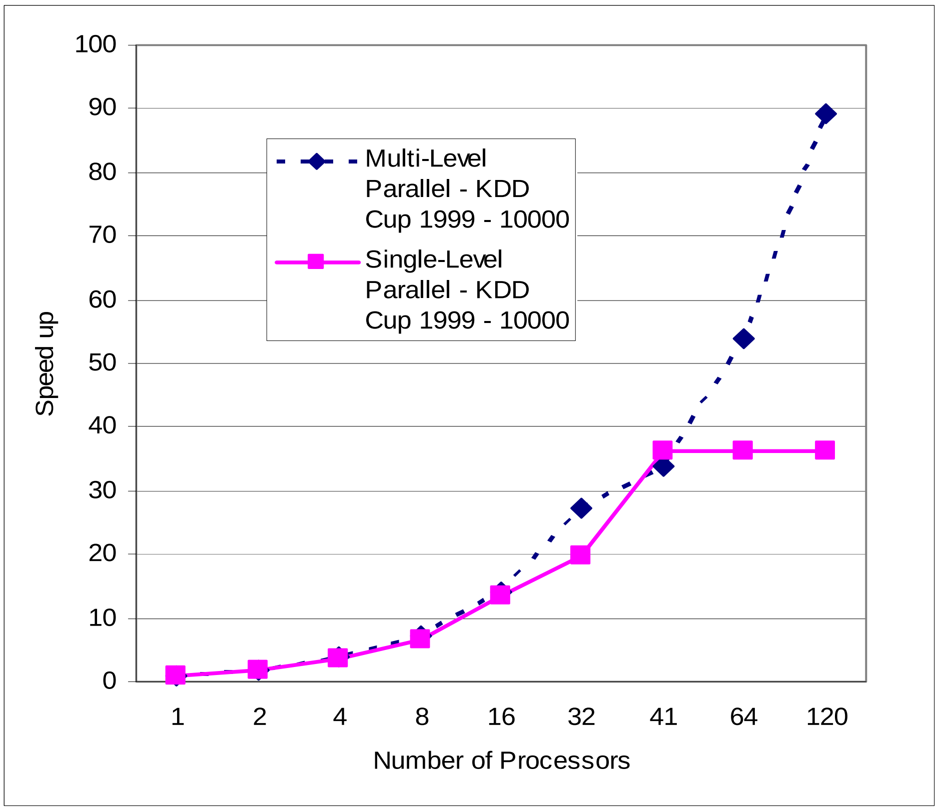

Figure 11 : Speed up of the two parallel algorithms with KDD Cup 1999 - 10000

5.4.3 *Efficiency*

Efficiency describes the advantage of the parallel algorithm. The efficiency is defined by speedup divided by the number of processors used:

$$\text{Efficiency} = \frac{\text{Sequential execution time}}{\text{Processors used} \times \text{Parallel execution time}} \qquad \textbf{(18)}$$

Figure 12 shows the efficiency of the proposed parallel algorithm with KDD Cup 1999 data set. We can see that the efficiency of both our parallel algorithms are quite good. However, the efficiency seems to be decrease when the number of processors is increase. Therefore, we can see that the transference data between processors and also the unbalancing takes lots time.

The transfer time is the time spends when the algorithms broad cast data and when receive the results by MPI_MIN to choose the best performance results from all processors. Besides the time spend for transfer data, there is also the time spend for waiting (unbalance). The reason is that, with difference value of input parameters (i.e. the thresholds and the number of cluster will be determined), the number of N-ESC sets will be very different. Therefore, the time for re-partitioning step is also become very different. It makes jobs on this processor may be finish before the jobs on the other processor. However, our methods to minimize the different number of jobs between processors in both the two parallel algorithms help much in decrease the unbalance.

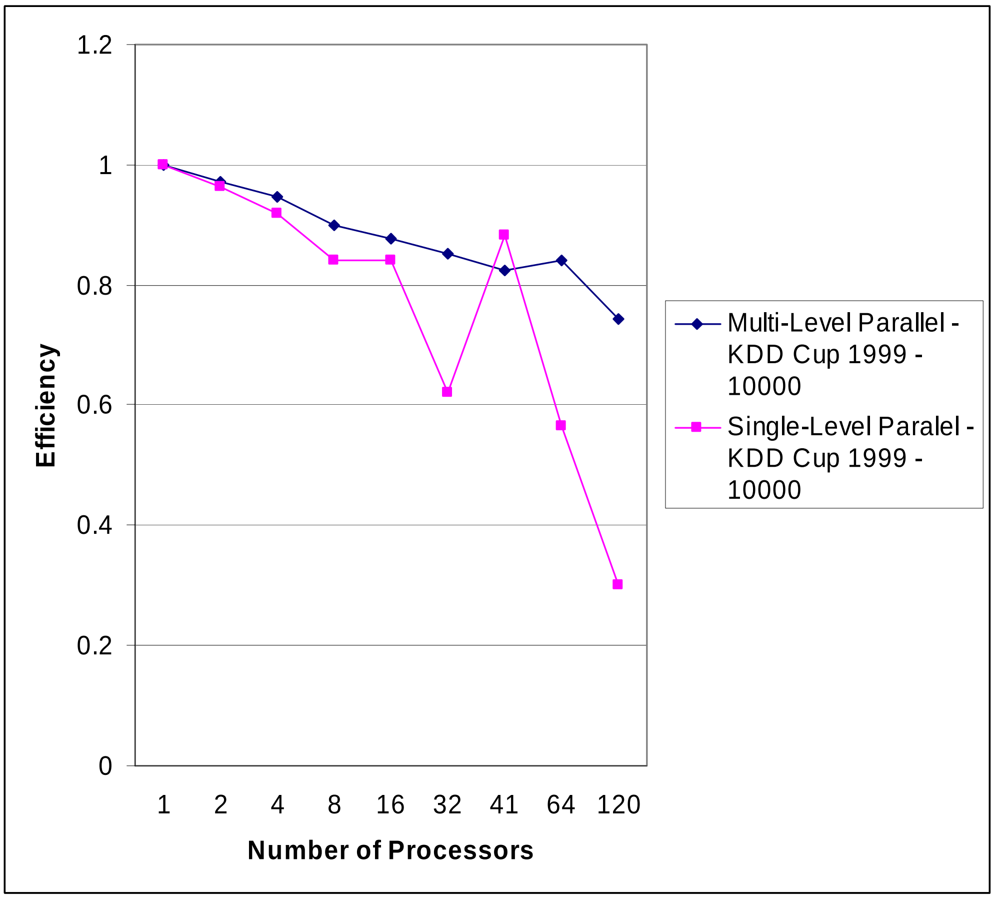

Figure 12: Efficiency of the two parallel algorithms with KDD Cup 1999 - 10000

5.5 Summary

In this chapter, evaluate the performances of two parallel algorithms. The Multi-Level Parallel algorithm seems to be much better than the Single-Level Parallel algorithm in case of processing on large enough number of processors with mixed data.

The average execution time, speed up, efficiency show that our proposed algorithms are effectiveness and scalable, especially the Multi-Level Parallel algorithm. The Multi-Level Parallel algorithm faster than the serial HAC algorithm approximately 90 times on 120 processors. It is a expected results for many parallel tasks.

Chapter 6. Conclusion and Future Work

6.1 Conclusion

In the thesis, we proposed a method to improve the original k-sets clustering algorithm [34] for categorical and mixed data. The improved algorithm named as High Accuracy Clustering Algorithm for Categorical and Mixed Data based on a new Grouping Approach. We also have proposed two parallel algorithms for the HAC algorithm. The first is Single-Level Parallel HAC algorithm and the remainder is Multi-Level Parallel HAC algorithm.

The idea for improvement arose from the desire to end up having objects connected with each other in the same class. To achieve this purpose, the idea of finding a class for each remaining object was replaced by the idea of finding a class for each remaining set which contains objects connected with each other.

The proposed improved algorithm was implemented on a Window OS, Pentium 4, CPU 2,53GHz. The experiments showed the better results on a clustering task than the original k-sets, and consequently, the results are better than those of previous works with certain databases.

Besides, to deal with large databases and multi-dimensional data, parallel techniques were employed to propose two effective and scalable parallel algorithms. It is realized that the time complexity for the improved algorithm depends on the number of attributes, in other words, it is dependant on the number of trials with the threshold θ. Therefore, one parallel approach can be developed based on the idea of dividing the tasks with a number of trials of θ equally over the processors. In theory, then, the complexity decreases P times compared with the serial improved algorithm.

As we see, the mixed data is quite popular in real database. In that case, the input parameters also consists of the threshold μ [34]. Consequently, the processing time will increase much depend on the trial of μ. We using multi-level parallel techniques to draw the Multi-Level Parallel algorithm for parallelize with both the trial of μ and θ.

The proposed parallel clustering algorithms for categorical and mixed data based on improving k-sets was implemented on a Cray T3E. The parallel algorithms was parallelized from the improved algorithm, therefore, the results in the clustering task were also better than the previous works as the results of the algorithm improved from k-sets. In addition, the Multi-Level Parallel algorithm reduces processing time significantly and can run approximately 90 times faster than the HAC algorithm on 120 processors with KDD Cup 1999 database. The proposed parallel HAC algorithms can deal with the clustering task and obtain useful knowledge from very large databases and multi-dimensional database in a reasonable time. They are showed as effective and scalable parallel algorithms.

6.2 Future Work

Continue to improve the re-partitioning step to get clustering algorithm with highest accuracy results for all kind of structure of databases.

In the future, new parallel techniques and a new similarity measure will be invented in order to obtain a more effective and efficient clustering algorithm.

Bibliography

[1] M. R. Anderberg. *Clustering Analysis for Applications*, Academic Press, 1973.

[2] G.H. Ball and D.J. Hall. Some fundamental concepts and sysnthesis procedures for pattern recognition preprocessors. In *International Conference on Microwaves, Circuit Theory, and Information Theory*, 1964.

[3] P. Berkhin. *Survey of Clustering Data Mining Techniques*, 2002.

[4] C.L. Blake and C.J. Merz. *UCI* Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science, 1998.

[5] P.S. Bradley, U.M. Fayyad. *Refining Initial Points for K–Means Clustering*. Proc. 15th International Conf. on Machine Learning, 1998.

[6] P. Cheeseman and J. Stutz. Baysian classification (autoclass): Theory and results. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. AAAI/MIT Press, 1996.

[7] U. M. Fayyad, G.Piatetsky-Shapiro, P.Smyth, R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press, ISBN 0-262-56097-6, 1996.

[8] D. H. Fisher. *Knowledge Acquisition Via Incremental Conceptual Clustering*, Machine Learning, 2(2), pp.139-172.

[9] D. Foti, D. Lipari, C. Pizzuti and D. Talia. *Scalable Parallel Clustering for Data Mining on Multicomputers*. In Proceedings of theWworkshop on High Performance Data Mining, IPDPS 2000, LNCS Volume 1800, pages 390-398. Springer Verlag, May 2000.

[10] K. Fukunuga. *Introduction to Statistical Pattern Recognition.* Acedamic Press, 1990.

[11] B. C. M. Fung. *Hierarchical document clustering using frequent itemsets*, A thesis submitted in partial fulfillment of the requirements for the degree of master of science in the information school of Computing Science, Simon Fraser Unirversity, 2002.

[12] V. Ganti, J. Gehrke, R. Ramakrishnan. *CACTUS: Clustering categorical data using summaries.* In Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD), 73–83, USA, 1999.

[13] D. Gibson, J.M. Kleinberg, P. Raghavan. *Clustering categorical data: An approach based on dynamical systems*. In Proceedings of the 24th International Conference on Very Large Data Bases, (VLDB), 311–322, USA, 1998.

[14] D.Q. Goodall. *A New Similarity Index Based On Probability*, Biometrics, *vo*l. 22, pp. 882-907, 1966.

[15] J. Grabmeier, A. Rudolph. *Techniques of Cluster Algorithms in Data Mining*, Data Mining and Knowledge Discovery, 6, pages 303-360, 2002 Kluwer Academic Publishers, Manufactured in The Neterlands, 2002.

[16] S. Guha, R. Rastogi, K. Shim. *CURE: An efficient clustering algorithm for large databases.* In Proc. of 1998 ACM-SIGMOD Int. Conf. on Management of Data, 1998.

[17] S. Guha, R. Rastogi, K. Shim. *CURE: ROCK: A Robust Clustering Algorithm for Categorical Attributes.* In Proc. of the 15th Int'l Conf. on Data Eng., 1999.

[18] E.H. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering based on association rule hypergraphs. Technical Report TR-97-019, Department of Computer Science, University of Minnesota, Minneapolis, 1997

[19] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers , 2001.

[20] S. Hettich, S. D. Bay. *The UCI KDD Archive* [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science, 1999.

[21] Z. Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In Research Issues on Data Mining and Knowledge Discovery, 1997.

[22] Z. Huang, ``Clustering large datasets with mixed numeric and categorical value''. Inprocedings of the First Pacific Asia Knowledge Discovery and Data Mining Conference World Scientific, Singapore, pp.21-34, 1997.

[23] Z. Huang, Michael K. Ng. A fuzzy *k*-modes algorithm for clustering categorical data, *IEEE Trans. Fuzzy Systems* 7 (4), 446–452, 1999.

[24] A.K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[25] R.A. Jarvis and E.A. Patrick. Clustering using a similarity measure based on shared nearest neighbors. *IEEE Transactions on Computers*, C-22(11), November 1973.

[26] M. V. Joshi, E-H. Han, G. Karypis, V. Kumar. *Paralell Algorithms for Data Mining*. In J.Dongarra, I. Foster, G. Fox, K. Kennedy, and A. White, editors, *CRPC Parallel Computing Handbook*, Morgan Kaufmann, 2000.

[27] D. Judd, P. McKinley, and A. Jain. Large-Scale Parallel Data Clustering. P*roceedings of the Int. Conf. on Pattern Recognition*, 1996.

[28] D. Judd, P. McKinley, A. Jain. Performance Evaluation on Large-Scale Parallel Clustering in NOW Environments. *Proceedings of the Eight SIAM Conf. on Parallel Processing for Scientific Computing*, Minneapolis, March 1997.

[29] D. Judd, P. McKinley, A. Jain. Performance Evaluation on Large-Scale Parallel Clustering in NOW Environments. *Proceedings of the Eight SIAM Conf. on Parallel Processing for Scientific Computing*, Minneapolis, March 1997.

[30] S. Kantabutra, Alva L. Couch. *Parallel K-means Clustering Algorithm on NOWs*. NECTEC Technical journal, Vol 1, No. 6, pages 243-248, January-February 2000.

[31] G. Karypis and V. Kumar. METIS 4.0: Unstructured graph partitioning and sparse matrix ordering system. Technical report, Department of Computer Science, University of Minnesota, 1998. Available on the WWW at URL *http://www.cs.umn.edu/˜metis*.

[32] G. Karypis and V. Kumar. A fast and highly quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1), 1999. Also available on WWW at URL http://www.cs.umn.edu/˜karypis. A short version appears in Intl. Conf. on Parallel Processing 1995.

[33] L. Kaufman and P.J. Rousseeuw. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.

[34] S. Q. Le, T. B. Ho, ``A k-sets Clustering algorithm for categorical and mixed data", In Proceedings of the 6th SANKEN (ISIR) International Synposium, pp.124-128, 2003.

[35] J. MacQueen. *Some methods for classification and analysis of multivariate observation*. In Proceedings 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297, 1967.

[36] R. Michalski, R. Stepp. *Learning from Observatioin: Conceptual Clustering*. Machine Learning: An Artifical Intelligence Approach, pages 331-363, 1983.

[37] R. S. Michalski, R. E. Stepp. *Automated Construction of Classifications: Conceptual Clustering Versus Numerical Taxonomy*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 5(4), pp. 396-410, 1983.

[38] Michael K. Ng, Joyce C.Wong. Clustering categorical datasets using tabu search techniques. *Pattern Recognition* 35, 2783-2790, 2002.

[39] R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. In *Proc. of the 20th VLDB Conference*, pages 144–155, Santiago, Chile, 1994.

[40] C.F. Olson. Parallel Algorithms for Hierarchical Clustering. *Parallel Computing*, 21:1313-1325, 1995.

[41] J.J. Peňa, J.A. Lozano, P. Larraňaga. *An empirical comparison of four initialization methods for the K-means algorithm.* Pattern Recognition Letters 20, pages 1027-1040, 1999.

[42] H. Ralambondrainy. *A conceptual version of the k-means algorithm*. Pattern Recognition Letters 16, 1147– 1157, 1995.

[43] K. Stoffel and A. Belkoniene. Parallel K-Means Clustering for Large Data Sets. *Proceedings of the Euro-Par '99*, LNCS 1685, pp. 1451-1454, 1999.

[44] K. Stoffel and A. Belkoniene. Parallel K-Means Clustering for Large Data Sets. *Proceedings of the Euro-Par '99*, LNCS 1685, pp. 1451-1454, 1999.

[45] Y. Sun, Q. Zhu, Z. Chen. An iterative initial-points refinement algorithm for categorical data clustering. *Pattern Recognition Letters* 23, 875-884, 2002.

[46] K. Thearling. *An Introduce to Data Mining,* [http://www.thearling.com/text/dmwhite/dmwhite.htm]

[47] L. Ungar and D. Foster. *Clustering methods for collarborative filtering*. AAAI Workshop on Recommendatioin Systems, 1998.

Publications

1. Nguyen Thi Minh Hai and Horiguchi Susumu, “Parallel Algorithm of K-sets Clustering for Categorical and Mixed Data”, Proc. Of the Joint Conference of Hokuriku Chapters of IEE, Fukui, Japan, 2003.

2. Nguyen Thi Minh Hai and Horiguchi Susumu, “A Parallel K-sets Clustering Algorithm for Categorical and Mixed ta”, IS-RR-2004-05, Research report, JAIST, Japan, 2004.