Doctoral Dissertation

Study on Evolutionary Data Mining for Database Clustering

March, 2016

I Gde Putu Wirarama Wedashwara Wirawan

Graduate School of Science and Engineering,
Yamaguchi University

Abstract

Data analysis underlies many computing applications, either in a design phase or just as a part of decision support for other applications and purposes such as medical, stock exchange, weather forecast, etc. One of the key elements in data analysis is classification of data based on goodness-of-fit to a postulated model, or natural groupings (clustering) revealed through analysis. In many system, database classification analysis, still managed manually by human, therefore the quality of distributed databases highly depends on designer's skill and the maintenance is very hard.

Clustering is the unsupervised classification of patterns into groups. Database is one of the common inputs for clustering which is usually processed in the form of a matrix of attributes (features) and records (data). This dissertation is concerned with the optimization of database clustering using evolutionary computation and fuzzy database modeling, and proposes four rule based clustering algorithms. Rule based clustering is one of the solutions to provide automatic database clustering and interpretation of data patterns. Rule based clustering represents data patterns as rules by analyzing database structures on both of attributes and records. Each cluster is created by a rule pool where there are many rules which have similarity values to other rules in the internal cluster and dissimilarity values to the other rules in the external clusters. The advantage of rule based clustering is to focus on important attributes represented by rules(frequent attribute patterns), while conventional data based clustering considers all attributes. In other words, rule based clustering can deeply consider the characteristics of the target database.

The aim of the optimization proposed in this dissertation includes : improvement of clustering quality for large databases, making clusters with different capacity limitation, and on-line rule updating capability to handle data changes in databases. The optimization of database clustering is realized by evolutionary rule based clustering using genetic network programming (GNP).

GNP is an evolutionary optimization technique, which uses directed graph structures instead of strings in genetic algorithm or trees in genetic programming, which leads to enhancing the representation ability with compact programs derived from the reusability of nodes in a graph structure. In this dissertation, GNP is used to handle rule extraction from databases by analyzing the attributes and records. Clustering is processed by grouping rules into the clusters based on the similarity measurement. GNP have the advantage of evolutionary algorithm for data classification which are global search ability for high speed rule extraction and easiness to stop evolution process that guarantee practical time rule extraction.

The cluster performance is evaluated by the following two evaluation methods, the silhouette and accuracy rate. The silhouette provides a succinct graphical representation of how well each object lies within its cluster. In unsupervised learning, it is difficult to evaluate the clustering results because the correct answers cannot be obtained. However, by using silhouette, the clustering performance can be evaluated because silhouette can consider the both of distance between similar data and dissimilar data. Accuracy Rate shows the percentage of records whose cluster labels are correctly assigned. Accuracy Rate is useful for evaluating the clustering result of data with correct class labels.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The first chapter describes the background of this study, which is the optimization of database clustering for distributed database management systems. The optimization proposed in this dissertation includes: 1) improvement of clustering quality and capability to handle high dimensional databases, and 2) additional clustering problems such as: making clusters with different capacity limitation, and on-line rule updating to handle unbalanced number increase/decrease of data in databases. In this dissertation, optimization methods of database clustering are realized by evolutionary rule-based clustering using genetic network programming (GNP). This dissertation also applies fuzzy database modeling as a feature representation method to improve the clustering ability to handle high dimensional databases.

1.1 Clustering

Data analysis underlies many computing applications, either in a design phase or just as a part of decision support for other applications and purposes such as medical, stock exchange, weather forecast, etc. Data analysis procedures can be dichotomized as either exploratory or confirmatory, based on the availability of appropriate models for the data source. One of the key elements in data analysis is classification of data based on goodness-of-fit to a postulated model, or natural groupings (clustering) revealed through analysis.

Clustering is the unsupervised classification of patterns into groups. The clustering problem has been addressed in many contexts and the usefulness of clustering has been verified as one of the steps in exploratory data analysis. The usefulness of clustering can be found in several exploratory pattern-analysis, grouping and decision-making [1].

Cluster analysis is the organization of a collection of patterns, which is usually represented by a vector of measurements, or a point in a multidimensional space belonging to a certain cluster based on the similarity (distance) between the point and the cluster.

1.1.1 Database Clustering

Computerized database was firstly introduced in 1960, where accesses can be done by single computer. In the 1970, computerized database was extended to centralized database which can be accessed by multiple computers via network.

Clustering can be applied to databases which is commonly processed in the form of matrix of attributes (features) and records (data) [1]. Database clustering is usually carried out to enhance the efficiency of database management systems.

The process of database clustering commonly includes feature extraction, pattern similarity measurement and clustering or grouping. Feature extraction is the use of one or more transformations of input features to produce new salient features or frequent pattern sets [2]. Complexity of the feature extraction increases as the complexity of database structures including variation of data and the number of attributes and records increases.

Typical database clustering activity involves the following steps [1]:

1. pattern representation (optionally including feature extraction and/or selection),
2. definition of a pattern proximity measure appropriate to the data domain,
3. clustering or grouping

The pattern representation refers to the number of classes, the number of available patterns, and the number, type and scale of the features being available to the clustering algorithm. In this study, GNP is used to execute feature extraction with rule extraction that will be explained in chapter 5.

The pattern proximity is usually measured by a distance function defined on pairs of patterns. A variety of distance measures are in use in the various communities [2, 3]. Euclidean distance can often be used to reflect dissimilarity between two patterns, whereas other similarity measures can be used to characterize the conceptual similarity between patterns [4].

The grouping step can be performed in a number of ways. The generated cluster(s) can be either hard (a definite partition of data into groups) or fuzzy (each pattern has a degree of membership in each cluster) [5].

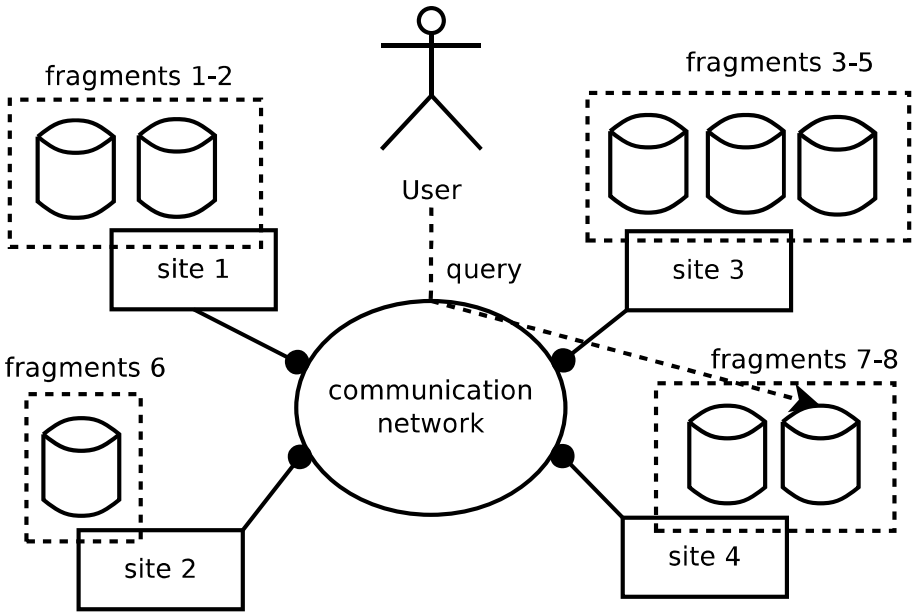

FIGURE 1.1: A distributed database environment

1.1.2 Rule Based Clustering

Rule based clustering is one of the solutions to provide automatic database clustering and interpretation of data patterns. Rule based clustering represents data patterns as rules by analyzing database structures on both of attributes and records. Each cluster is created by a rule pool where there are many rules which have similarity values to other rules in the internal cluster and dissimilarity values to the rules in the external clusters [6–8]. The advantage of rule based clustering is to focus on important attributes represented by rules(frequent attribute patterns), while conventional data based clustering considers all attributes. In other words, rule based clustering can deeply consider the characteristics of the target database.

1.1.3 Clustering in Distributed Database Management System

Distributed database management system (DDBMS) could be a solution for dealing with large scale information systems with large amount of data growth and data accesses. A distributed database (DDB) is a collection of data that logically belongs to the same system but is spread over the sites of a computer network (Fig. 1.1). A DDBMS is then defined as a software system that permits the management of DDB and makes the distribution of data between databases and software transparent to the users [9, 10].

To handle the data proliferation, efficient access methods and data storage techniques have become increasingly critical to maintain an acceptable query response time. One

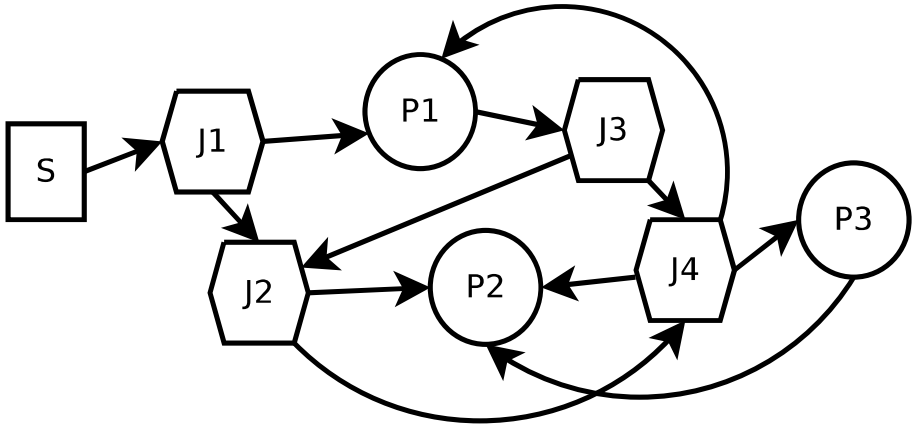

FIGURE 1.2: Basic Implementation of GNP

S : start node, [J_1,...,J_4] : judgment node, [P_1,...,P_3] : processing node

way to improve query response time is to reduce the number of disk I/Os by clustering the database vertically (attribute clustering) and/or horizontally (record clustering) [11, 12]. Improvements in the retrieval time of multi-attribute records can be attained if similar records are grouped close together in the file space as a result of restructuring. For example, fewer page transfers would occur if two or more of the target records reside in the same page [13]. The distribution rules of data of the conventional DDBMS is basically made by human, therefore the quality of distributed databases highly depends on designer's skill and the maintenance is very hard. The DDBMS proposed in this dissertation aims to realize automatic database management where distribution of data is executed by data mining, the maintenance of databases is easy and the extension to fuzzy database is possible.

1.2 Genetic Network Programming

GNP is an evolutionary optimization technique, which uses directed graph structures instead of strings in genetic algorithm [14] or trees in genetic programming [15], which leads to enhancing the representation ability with compact programs derived from the re-usability of nodes in a graph structure.

1.2.1 Basic Application of GNP

In GNP, there are two types of nodes: judgment nodes and processing nodes. A judgment node has a function to examine a value of attribute and select Yes/No branches. A processing node is used to show the start position of the node transition. Therefore,

the node transition starts from one of the processing nodes, and the sequence of nodes represents rules of the program.

The basic structure of GNP is illustrated in Fig. 1.2, showing the directed graph consisting of judgment nodes and processing nodes. p ($p = 1, \ldots, n$) denotes the p-th judgment function stored in a library for judgment nodes, while q ($q = 1, \ldots, m$) denotes the q-th processing function stored in a library for processing nodes [16, 17].

1.3 Rule Extraction by GNP

In this study, GNP is used to implement rule extraction from databases by analyzing the records. Each judgment node represents an attribute with value range. For example, price attribute could be divided into three ranges (low, middle, high), and one range is assigned to one judgment node. GNP makes rules by evolving combinations of nodes and measures the coverage of the extracted rules. Coverage means how much the records in a database each rule can represent (cover),and rules covering at least one record will be stored in a rule pool. The clustering of this study is realized by distributing the stored rules to several sites (clusters). The point of the clustering proposed in this study is to distribute rules, not the data, which contributes to distributing any data into sites considering the similarities between rules and data.

The rule extraction of GNP is executed analyzing the database information such as:

Attributes amount : the number of attributes in a database. Each attribute will be divided into some nodes depending on its variation and value ranges (distance of minimum value and maximum value).

Data amount : the number of records in a database.

Data variation : how much different records are contained in a database. If every record in a database is different, variation is 100%, if half of the records in the database is different, variation is 50%, and if every record in a database is the same, variation is $1/(the number of data) \times 100\%$. For example, in Table 2.1 there are six data variation in total 310 data, thus the variation is $(6/300) \times 100 = 1.94\%$.

GNP is used to extract rules from a database by analyzing all the records. Phenotype and genotype structures of GNP are described in Fig. 1.3 and Table 1.2, respectively. In Fig. 1.3, each node has its own node number (1–11), and in Table 1.2, the node information of each node number is described. The program size depends on the number of nodes, which affects the amount of rules created by the program.

TABLE 1.1: Example of Frequency Table of Price Attribute

x	f	xf
10	30	300
25	25	625
50	30	1500
80	140	11200
100	65	6500
150	20	3000
Total	310	23125

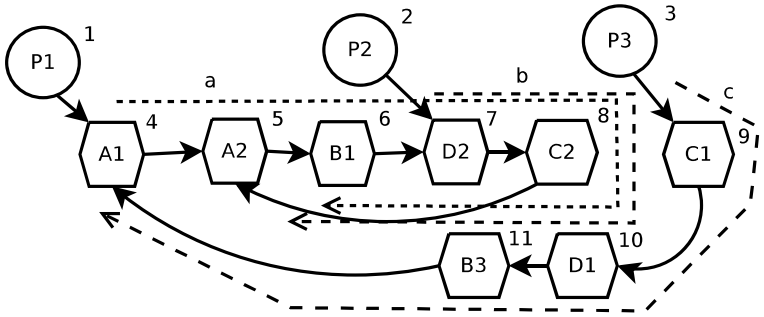

FIGURE 1.3: GNP Implementation on Data Mining

TABLE 1.2: GNP gene structure of Fig. 1.3

i	NT_i	A_i	R_i	C_i
1	1	0	0	4
2	1	0	0	7
3	1	0	0	9
4	2	A	1	5
5	2	A	2	6
6	2	B	1	7
7	2	D	2	8
8	2	C	2	5
9	2	C	1	10
10	2	D	1	11
11	2	B	3	4

i : Node number,

NT_i : Node types; 1=processing, 2=judgment,

A_i : Attribute index,

R_i : Attribute range index,

C_i : Connection

In the implementation of rule extraction, a judgment node represents an attribute of the database, which is represented by A_i showing an attribute index such as price, stock, etc., and R_i showing a range index of the attribute value. For example, $A_i = A$ represents price attribute, and $R_i = 1$ represents value range $[0, 50]$ and $R_i = 2$ represents value range $[51, 80]$. A processing node show the start point of the sequence of judgment nodes which are executed sequentially by their connection. Sequences of nodes starting

TABLE 1.3: Example of database

A_1	A_2	B_1	D_2	C_2	C_1	D_1	B_3
1	0	1	0	0	1	1	0
1	0	1	1	1	0	0	0
0	1	0	1	1	0	0	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
1	0	0	0	0	1	1	1

from each processing node (P_1, P_2, P_3) are represented by dotted line a, b and c. A node sequence flows until support for the next combination does not satisfy the threshold. The nodes with the attributes that have already appeared in the sequence will be skipped. Candidate rules extracted by the program of Fig. 1.3 to the database of Table 1.3 to the database of Table 1.4. In Table 1.4, three rules are extracted by the node sequence from each processing node.

The score of rule is defined as follows.

$$\text{Score of rule } r = \begin{cases} 0 \text{ if } sup(r) = 0 \\ sup(r) + (n_{con}(r) - 1) \text{ if } sup(r) > 0, \end{cases} \tag{1.1}$$

where $sup(r)$ is the support[1] of rule r and $n_{con}(r)$ is the length of rule r.

Fitness for evaluating an individual is defined as follows.

$$\text{Fitness} = \sum_{r \in R} \{sup(r) + (n_{con}(r) - 1) + \alpha_{new}(r)\}, \tag{1.2}$$

where $\alpha_{new}(r)$ is an additional value if rule r is newly extracted.

Table 1.4 shows the length and support of the extracted rules. Score of rule described by Eq. 1.1 is not only calculated by its support($sup(r)$) but also by its length($n_{con}(r)$). Considering the rule length makes rules more reliable because longer rules can cover various combinations of attributes. For example, $A_1 \wedge B_1$ has relatively high support 3/6 but only has the length two, so the score of rule is only 1.500. On the other hand, $C_1 \wedge D_1 \wedge B_3 \wedge A_1$ has the support only 1/6 but the length is four, therefore, the score

[1]Ratio of records that satisfy rule r

TABLE 1.4: Example of database and its support to the extracted rules

Processing Nodes	Extracted Rules	Support	Score	
			Rule	Template
1	$A_1 \wedge B_1$	3/6	1.500	6.00
	$A_1 \wedge B_1 \wedge D_2$	1/6	2.166	3.67
	$A_1 \wedge B_1 \wedge D_2 \wedge C_2$	1/6	3.166	4.67
2	$D_2 \wedge C_2$	2/6	1.166	4.33
	$D_2 \wedge C_2 \wedge A_2$	1/6	2.166	3.67
	$D_2 \wedge C_2 \wedge A_2 \wedge B_1$	0/6	0	0
3	$C_1 \wedge D_1$	2/6	1.333	4.33
	$C_1 \wedge D_1 \wedge B_3$	1/6	2.166	3.67
	$C_1 \wedge D_1 \wedge B_3 \wedge A_1$	1/6	3.166	4.67
		Total	19.99	

(Score of template is introduced in section 2.4.1.3)

becomes 3.166. $\alpha_{new}(r)$ is also included in the fitness because the objective of rule extraction is to discover new rules from a database as much as possible.

1.4 Fuzzy Database

Existing DBMS are basically able to handle crisp, precise and non-ambiguous data. These systems do not cater for vague and ambiguous data which is based on fuzzy. Fuzzy databases [18, 19] become one of the solutions to deal with uncertain or incomplete information using fuzzy logic that is represented by fuzzy queries. Integration of the fuzzy logic into distributed databases has advantages such as flexible querying, handling imprecise data, minimizing the transformation costs and the applicability to fuzzy data mining [20].

The data is sometimes transformed before being used. One reason for this is that different attributes may be measured on different scales [1, 21]. In cases where the ranges of values differ widely from attribute to attribute, these different scales of attributes would influence the results of the cluster analysis, and it is common to normalize the data so that all attributes are on the same scale. Another reason for initially transforming the data is to reduce the number of dimensions, particularly when the initial number of dimensions is large. In this study, fuzzy database modeling is used as fuzzy logic based standardization, which is explained in the next subsection.

1.5 Fuzzy Object Oriented Database (FOOD)

A general database consists of a collection of records stored in a computer storage. A fuzzy database is an extended structure of the general database where uncertain or incomplete information can be dealt with using fuzzy logic. Flexibility of fuzzy databases can be achieved by adding fuzzy membership degree to each record, that is, the membership degree of an attribute of each record is represented by the value range between 0 and 1 (not binary values like general databases). Fuzzy values of attributes are defined by fuzzy sets associated to each attribute and with different membership functions [22, 23].

Fuzzy object oriented database (FOOD) is a database with fuzzy membership extension for making an object-oriented database model that permits data values to be fuzzy predicates and numbers. FOOD considers the effects of imprecise data values on the class structures and proposes a modeling paradigm compatible with the object-oriented data model that accommodates uncertainty in class hierarchies. From a database perspective, this extension has a significant advantage: it provides more accurate description of the universe of discourse which is very important in knowledge intensive applications, such as the combinations of databases and artificial intelligence systems [24–27]. The model of FOOD adopted in this study consists of:

1. Class : name of object structure such as product, employee, etc.
2. Variable : component of class. For example, class “product” has a variable such as product’s name, price, weight etc.
3. Term : fuzzy membership function for each variable. For example, the price of class “product” has three terms such as cheap, average and expensive (explained later in section 3).
4. Query(rule) : detailed structure of fuzzy query [22]. Basically only SELECT query is used in this study. SELECT query covers target records in a database by the combination of fuzzy terms. Query will be the main object of GNP rule extraction, i.e., the core part of the proposed clustering algorithm.

1.6 Clustering Evaluation

The cluster performance is evaluated by the following two evaluation methods : Silhouette [28] and Accuracy. Silhouette provides a succinct graphical representation of how well each object lies within its cluster. Silhouette value is calculated by Eq. 1.3.

Because each attribute has its own value scale, it is normalized using mean and standard deviation before calculating Silhouette. Silhouette is useful for evaluating the clustering result without using correct class label.

$$\begin{aligned} s &= \frac{b-a}{\max\{a,b\}} \\ &= \begin{cases} 1-a/b, & \text{if } a < b \\ 0, & \text{if } a = b \\ b/a - 1, & \text{if } a > b \end{cases} \end{aligned} \tag{1.3}$$

s: Silhouette value for a single sample. The Silhouette value for a set of samples is given as the mean of the Silhouette values of each sample.

a: the average dissimilarity (distance) of data within the same cluster.

b: the lowest average dissimilarity (distance) to any other cluster.

Accuracy Rate shows the percentage of records whose cluster labels are correctly assigned. Accuracy Rate is useful for evaluating the clustering result of data with correct class labels.

1.7 Objective of this study

The objective of this study is to realize evolutionary database clustering and the detailed objectives in each chapter are described as follows.

1. Clusters with different capacity limitation can be generated by GNP that extracts a large number of rules and dynamic programming that solves knapsack problem.
2. On-line rule updating ability is realized, which enables rule based clustering using GNP to have adaptability to the unbalanced increase/decrease of the number of data in databases.
3. The concept of fuzzy object oriented database modeling is introduced to the rule based clustering using GNP to improve clustering quality and data representation abilities.
4. A fuzzy feature selection method is proposed to realize high clustering quality for high dimensional databases.

The other features are described as follows.

1. Deal with clustering of rules, not clustering of data.
2. Separate a single database to multiple databases by considering similarity of rules that extracted from single database.
3. Organize data by frequent patterns (rules) that occur in a single database. The data with similar frequent patterns are stored in the same storage.

The Contribution of this study are described as follows.

1. Propose new clustering algorithms that give additional mechanisms to rule-based clustering to realize practical and user-friendly systems.
2. Aim to obtain good clustering performance comparing to the conventional methods as well.

1.8 Structure of this dissertation

The outline of this study are described as follows.

1. Chapter 2 discusses implementation of genetic network programming (GNP) and standard dynamic programming to solve the knapsack problem (KP) as a decision support system for record clustering in distributed databases. This chapter also discusses partial random rule extraction method in GNP to discover frequent patterns in a database for improving the clustering algorithm, especially for large data problems. The concept of KP in clustering is discussed, which is to distribute rules to each site by considering similarity (value) and data amount (weight) related to each rule to match the site capacities.
2. Chapter 3 discusses decision support system for database cluster optimization using GNP with on-line rule updating based clustering. In this chapter on-line algorithm is utilized to maintain the cluster adaptability against several unbalanced data growth. To realize this ability, start node is added to represent the start positions of the node transition, and processing node which determines addition/deletion of rules and to which cluster each rule should be assigned is added to the conventional GNP based rule extraction method.
3. Chapter 4 discusses a clustering method using GNP with the advantages of fuzzy object oriented database (FOOD) modeling. The main purpose of this chapter is to provide additional mechanisms to database clustering systems, that is, a data

mining algorithm for extracting fuzzy rules, and building clusters based on the extracted fuzzy rules. The adoption of FOOD model to GNP rule extraction process can increase the clustering quality and interpretation of clustering structures.

4. Chapter 5 discusses a database clustering using GNP with feature selection and representation of fuzzy database. This chapter is an expansion of chapter 4 that aims for the improvement of clustering quality on high dimensional databases. Feature selection with fuzzy database is introduced in this chapter, where database is examined and many fuzzy membership functions are generated to calculate attribute relevancy. After the feature selection, attributes are grouped by considering their relevancy in the clustering process. Then, rule extraction is performed by GNP separately using each group of attributes.

5. Finally, chapter 6 makes conclusions to describe the main achievements of this study in the optimization of database clustering and its additional problems.

Chapter 2

Evolutionary Rule Based Clustering Using Combination of Genetic Network Programming and Knapsack Problem

2.1 Chapter Introduction

This chapter involves the implementation of genetic network programming (GNP)[16, 17] and standard dynamic programming to solve the knapsack problem (KP)[29, 30] as a decision support system for record clustering in distributed databases. Fragment allocation with storage capacity limitation problem is a background of the proposed method. The problem of storage capacity is to distribute sets of fragments into several sites (clusters). Total amount of fragments in each site must not exceed the capacity of site, while the distribution process must keep the relation (similarity) between fragments within each site.

The objective is to distribute big data to certain sites with the limited amount of capacities by considering the similarity of distributed data in each site. To solve this problem, GNP is used to extract rules from big data by considering characteristics (value ranges) of each attribute in a database. The proposed method also provides partial random rule extraction method in GNP to discover frequent patterns in a database for improving the clustering algorithm, especially for large data problems. The concept of KP is used to distribute rules to each site by considering similarity (value) and data amount (weight)

related to each rule to match the site capacities. Therefore, The proposed method provides a new clustering method with additional storage capacity problem. The clustering performance is evaluated by the simulations using benchmark databases and compared to conventional clustering algorithms.

This chapter is organized as follows. Section 2.2 describes a review of the proposed framework, section 2.3 describes the detailed algorithm of the proposed framework, section 2.4 shows the simulation results, and finally section 2.5 is devoted to conclusions.

2.2 Review of the Proposed Framework

KP is a combinational optimization problem dealing with a set of items, each with a mass and a value, determining the number of each item to include in a collection so that the total weight is less than or equal to the given limit and the total value is as large as possible. KP is defined as follows.

$$\text{maximize } S = \sum_{i=1}^{n} v_i x_i, \quad \text{subject to} \quad \sum_{i=1}^{n} w_i x_i \leq W, \tag{2.1}$$

where S = total value of the knapsack (site); i = fragment number ($1 \leq i \leq n$); x_i = the number of fragments i; v_i = value (similarity to the leader rule of the site) of fragment i; w_i = weight (data size) of fragment i; W = capacity of the site. By allowing each fragment (item) to be added more than once to sites, this optimization can handle the problem of replication [29, 31].

Knapsack problem in this study is solved by standard dynamic programming for 0/1 knapsack problem [32]. Let us define two dimensional array $m[i, w]$ with row i and column w. $m[i, w]$ shows the value of knapsack when considering items with item number $1, 2, \ldots, i-1, i$, and their total weight w. $m[i, w]$ is calculated by Eq. 2.2.

$$\begin{aligned} m[i,\ w] &= m[i-1,\ w] \quad \text{if} \quad w_i > W \\ m[i,\ w] &= \max(m[i-1,\ w],\ m[i-1, w-w_i] + v_i) \quad \text{if} \quad w_i \leqslant W. \end{aligned} \tag{2.2}$$

The first step is to calculate $m[0, w]$, then $m[1, w]$ is calculated based on the values of $m[0, w]$. The same process is repeated to calculate $m[2, w], \ldots, m[n, w]$. After finishing calculating $m[i, w]$, the maximum value among all $m[n, w]$ ($0 \leq w \leq W$) is selected as a solution of the problem.

In this study, standard dynamic programming is applied to solve the KP to handle a distribution of rules extracted by GNP to each site. Rules with high data coverage will be the leaders of each site and KP will consider the similarity between the leader rules and remaining rules (similarity is considered as a value of item (rule) in KP), and also consider the coverage of rules (coverage is considered as weight in KP) to match with site capacities. Therefore, the similar rules to a certain leader are basically put into the same site.

The unique point of this study is to regard this problem as putting similar data into the same servers, but not making the total amount of data exceed the storage capacity. In other words, the similarity of data is regarded as the value of objects, and the data frequency is regarded as the weight, which is the new idea that has never done before.

2.3 Literature review

The proposed method aims to realize rule based clustering, where GNP is used for rule extraction that has been proposed in [16], then standard dynamic programming is used to solve KP in the storage capacity problem of fragment allocation in distributed databases that has been introduced in [33]. Introducing storage capacity problem to the database clustering and introducing the concept of KP to solve the problem is one of the unique points of the proposed method. Moreover, the proposed method provides partial random feature selection in the rule extraction, which can discover frequent patterns of attributes in a database and improve the clustering quality. With the above features, the proposed method provides an automatic record clustering that aims to be a decision support system for record clustering in distributed databases.

This study involves the implementation of genetic network programming (GNP) for data mining and standard dynamic programming to solve the knapsack problem (KP) for the rule based clustering. Introducing storage capacity problem to the database clustering and introducing the concept of KP to solve the problem is one of the unique points of the proposed method. Moreover, the proposed method provides partial random feature selection in the rule extraction, which can discover frequent patterns of attributes in a database and improve the clustering quality. With the above features, the proposed method provides an automatic record clustering that aims to be a decision support system for record clustering in distributed databases.

The current related literature about fragment allocation is [34]. The study presents an approach which simultaneously makes data fragments vertically and allocates the fragments to appropriate sites across the network. Bond Energy Algorithm (BEA) is

applied with a better affinity measure that improves the quality of the generated clusters of attributes. BEA can find good relations between attributes by discovering frequent items between records in a database. The proposed method also discovers frequent pattern sets to perform an automatic horizontal fragmentation or record clustering, not a vertical fragmentation as proposed by this literature.

The current related clustering topic is an automated feature weight learning proposed by [35]. This article presents and investigates a new variant of the fuzzy k-Modes clustering algorithm for categorical data with automated feature weight learning. This method automatically associates higher weights to features which are instrumental in recognizing the clustering patterns of the data in the classical fuzzy k-Modes algorithm. The proposed method in this chapter also discovers frequent pattern sets of features (attributes) to improve the performance of clustering, which is explained in section 4.1.3, and moreover, the proposed method can deal with a storage capacity problem that has not been solved in this literature.

Another related topic is evolutionary fine-tuning of automated semantic annotation systems proposed by [36]. The literature proposes a Parameter Tuning Architecture (PTA) for automating the task of configuring parameter values of semantic annotation tools with evolutionary computation. The similarity with the proposed method is the usage of evolutionary computation to find the proper combinations of features for solving the problem and use feature weight selection, but the problem of this literature, i.e., semantic annotation, is different from the proposed method in this chapter, i.e., the target problem of the proposed method is a record clustering with additional storage capacity limitation problem.

2.4 Combination of GNP and Knapsack problem

The implementation for processing record clustering is separated into two parts: GNP rule extraction and KP rule distribution.

2.4.1 GNP Rule Extraction

The node preparation for GNP rule extraction contains two phases: node definition and node arrangement. In addition, two kinds of node arrangement methods are proposed: one is full random arrangement and the other is partial random arrangement.

TABLE 2.1: Example of Frequency Table of Price Attribute

x	f	xf
10	30	300
25	25	625
50	30	1500
80	140	11200
100	65	6500
150	20	3000
Total	310	23125

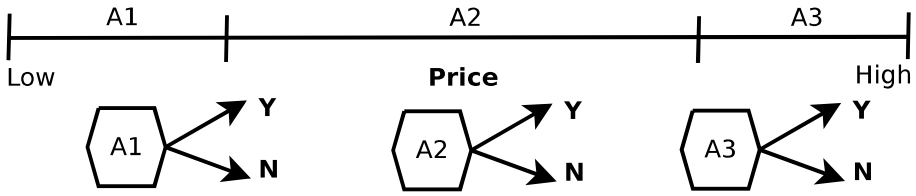

FIGURE 2.1: Node for judging attributes

2.4.1.1 Node Definition

The main purpose of node definition is to preparing judgment nodes that will be combined to create rules. First step is to find the minimum and maximum values of each attribute. For example, the minimum value of "price" attribute is 10 and the maximum value is 150 in the database with 310 records. Then, a frequency table is created per attribute as shown in Table 2.1. x shows the price of a product, and f shows how many times the product with the same price is recorded in the database. For example, product(s) with price $x = 10$ appeared 30 times. Then, mean value of $(\overline{xf})$ is calculated by Eq. 2.3.

$$\overline{xf} = \frac{\sum xf}{\sum f} = 74.60 \tag{2.3}$$

To define nodes from Table 2.1, data should be divided equally based on the amount of data. For example, three nodes could be created by dividing value range into three ranges considering the occurrence frequency as shown in Fig. 2.1. In this example, three ranges are: $x = \{10, 25, 50\}$ (85 data), $x = \{80\}$ (140 data) and $x = \{100, 150\}$ (85 data). First node and third node contain more than one price because each single record (10,25,50,100,150) does not have enough frequency to be defined as node. Mean $(\overline{xf} = 75.42)$ is used to measure the minimum coverage to become a node. Through the

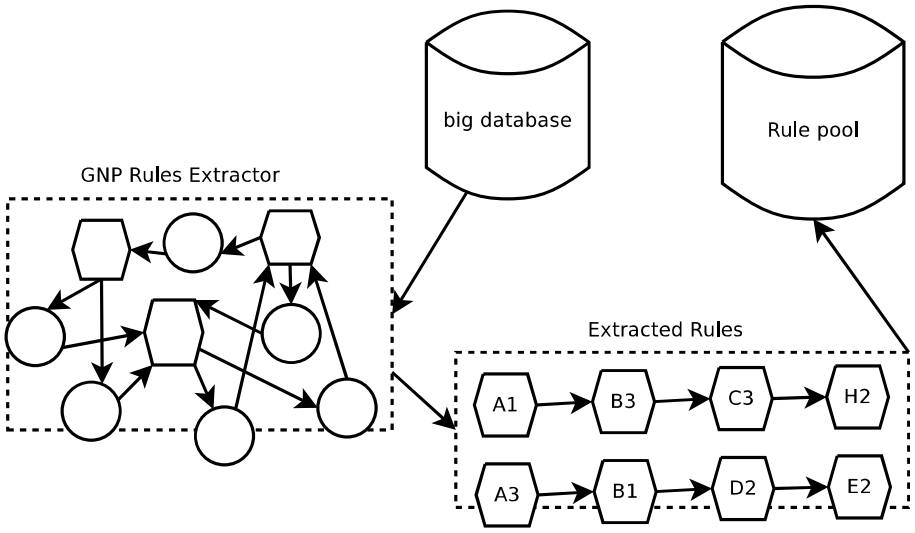

FIGURE 2.2: GNP rule extraction

measurement, the second node can be created from single record ($x = \{80\}$) because $f = 140$ exceeds $\overline{xf}$.

2.4.1.2 Node Arrangement : Full Random

The purpose of node arrangement is to select necessary nodes for efficiently extracting a large number of rules. Full random method randomly selects nodes from the defined nodes in section 2.4.1.1 and makes graph structures. From the created graph structures, GNP extracts a large number of important rules and stores them in the rule pool (Fig. 2.2). The original framework of the rule extraction is described in [17] in detail.

After rules are extracted, GNP will measure the amount of coverage archived by the rules. In this study, coverage of rule r means the number of records that match (covered by) rule r. If a rule covers at least one data, such rule is added to a rule pool, otherwise, the rule is discarded. Rules with high coverage will be defined as elite rules and be the leaders of each cluster (site) in KP process. Rule extraction process continues until all the records in a database are covered.

To create a large number of good rules, crossover and mutation are executed.

Crossover: exchange one or more node(s) between parents to make new rules

Mutation: change one or more node(s) to make different combination of nodes

Crossover is effective to switch weak nodes (nodes with less data frequency) of the parents with strong nodes (nodes with more data frequency). Mutation is effective to switch weak nodes of one individual to strong nodes.

2.4.1.3 Node Arrangement : Partial Random

Partial random method has two sequential processes of GNP the first process is to find template rules and the second process is to execute general rule extraction of GNP combined with the templates created in the first process. Templates are extracted to obtain combinations of attributes that frequently happen in the database. Score of template is calculated by Eq. 2.4, and the templates with high scores will be used in the second process.

$$\text{Score of template } t = \begin{cases} 0 \text{ if } sup(t) = 0 \\ 10 * sup(t) + (n_{con}(t) - 1) \text{ if } sup(t) > 0 \end{cases} \tag{2.4}$$

Contrary to the score of rule (Eq. 1.1) which gives more weight on the node length, the score of template gives more weight on support as shown by Eq. 2.4. For example, the scores of templates are shown in Table 1.4 where the results are relatively contrast to the score of rules. $A_1 \wedge B_1$ has the highest score of template although the node length is just two. When $A_1 \wedge B_1$ is used as a template, partial random will be implemented by randomizing remaining attributes such as C and D.

In the template extraction process, only a few number of attributes are included in GNP rule extraction. It aims to increase the possibility to get templates with high support. For example, in "A. finding template" in Fig. 2.3, the combination of attribute A and D is defined as a template as a result of the score calculation (Eq. 2.4). It will increase the possibility to find good combinations with attribute A and D. In "B. rule extraction", the template and the remaining attributes, that is B and C, are considered. The rule extraction process can obtain rules with longer length than the templates.

Table 2.2 shows a simple example of partial random for easy explanation. Each template contains attribute A and D, and it is combined with the remaining attributes, that is B and C. The generated rule of $A_3 \wedge D_3 \wedge B_1 \wedge C_2$ obtains the highest score of rule (Eq. 1.1) because it has long rule length and high coverage.

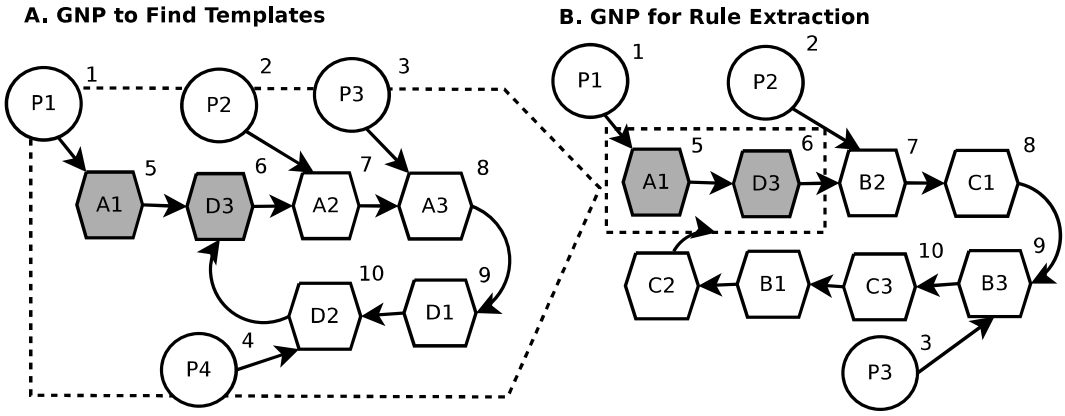

FIGURE 2.3: Node Arrangement Optimization in GNP

TABLE 2.2: Example of combination of templates with remaining attributes

Template	Remaining attributes	Coverage	Score of rule
$A_2 \wedge D_3$	$B_1 \wedge C_2$	0	0
$A_2 \wedge D_3$	$B_3 \wedge C_2$	10	40.4
$A_1 \wedge D_3$	B_3	24	34.5
$A_3 \wedge D_3$	$B_1 \wedge C_2$	14	40.5

2.4.2 Rule Distribution based on the standard dynamic programming for solving Knapsack Problem

After all the records in a database are covered by rules extracted by GNP, standard dynamic programming for solving KP problem is used to distribute rules to several sites. Rules with high coverage (elite) become the leaders of each site, then application considers the similarity of the remaining rules to the leader rules (value) and coverage of the rules (weight) are considered to distribute the remaining rules to the sites. Similarity of remaining rule to the leader rules is calculated by Eq. 2.5.

$$S(r_1, r_2) = \frac{N_{match}(r_1, r_2)}{Max\{N_{ante}(r_1), N_{ante}(r_2)\}} \tag{2.5}$$

$S(r_1, r_2)$: similarity between rule r_1 and r_2, $N_{match}(r_1, r_2)$: the number of matched attributes between r_1 and r_2, $N_{ante}(r)$ $(r \in \{r_1, r_2\})$: the number of attributes in rule r.

$Max\{N_{ante}(r_1), N_{ante}(r_2)\}$ means that longer rule length becomes a divider to the number of matched attributes between two rules $(N_{match}(r_1, r_2))$. When the longer rule

TABLE 2.3: Example of similarity calculation between leader and remaining rules

Rule	A	B	C	D	$N_{match}(r_1, r_2)$	$S(r_1, r_2)$
Leader	A_1	B_3	C_2	-	-	-
1	$*A_1$	B_2	C_1	$*D_2$	2	2/4
2	A_2	$*B_3$	$*C_2$	$*D_1$	3	3/4
3	$*A_1$	B_1	$*C_2$	-	2	2/3

* : matched attribute

includes attributes that are not contained in the shorter rule, those attributes are assumed to be matched. Examples of similarity calculation are shown in Table 2.3. From Table 2.3, rule 2 shows the highest similarity to the leader. The leader rule does not have attribute D, so every attribute D in the remaining rules is assumed to be matched.

2.4.3 Complexity Analysis

The main processes of the proposed method with their complexity analysis are explained as follows :

1. Rule extraction part
   (a) Node definition : This process prepares judgment nodes that will be combined to create rules. Complexity in this process is related to the number of data and attributes. The large number of attributes affects the number of nodes to be defined. The large number of data affects the complexity of creating a frequency table per attribute.
   (b) Node arrangement : This process selects necessary nodes for efficiently extracting a large number of rules. Complexity in this process is related to the number of attributes. The large number of attributes affects the number of possible combinations of attributes that could be extracted. Rule extraction process continues until all the data in a database are covered, therefore, the large number of possible combinations requires more iterations to cover all the data. To efficiently dealing with this complexity, partial random method is designed to hold the frequent patterns with high coverage to be used in next iteration.
   (c) Extracted rules measurement : This process measures the coverage archived by the extracted rules. Complexity in this process is related to the number of data. The large number of data affects the number of measurement process of each rule.

2. Rule distribution part: Standard dynamic programming is used to solve the KP problem. that is, the extracted rules are distributed to several clusters with the consideration of the similarity between rules (value) and coverage of the rules (weight). Each cluster cannot store all the rules when the sum of the coverage of the rules exceeds the storage limitation. Complexity in this process is related to the number of rules and clusters, and the storage limitations of each cluster. The large number rules increases the complexity by increasing the possible combinations of the rule distribution, while the large number of clusters and small storage limitations also increase the complexity by compounding the several purposes of distribution process.

Basically, the proposed method takes more time to complete making clusters than the conventional clustering methods because it require more time for initialization and evolutionary process. However, the proposed method can extract rules from huge databases in a practical time, which is the advantage over the conventional rule extraction method, e.g., Apriori method. In addition, the proposed method can stop extracting rules at any time (any generation) when sufficient number of rules are obtained. This flexibility is also the advantage of the proposed method.

2.5 Simulations

First, full random and partial random methods in the rule extraction of GNP are compared. Then, knapsack rule distribution is carried out and its results are compared with other five methods.

The six datasets are downloaded from UCI machine learning repository (shown in Table 2.4) for the comparison, and the clustering performance is evaluated by both Silhouette value and accuracy rate. For the clustering performance evaluation, this study uses the benchmark datasets with the predefined number of clusters. In real world problems, the number of clusters is unknown, so it will be the important future study.

2.5.1 Simulations of GNP Rule Extraction

The simulations of GNP rule extraction are separated into the comparisons of crossover and mutation rate to obtain optimal parameter setting and the comparison of node arrangement methods.

TABLE 2.4: UCI database

	Attribute	Classes	Samples	Data Type
Wine Quality	12	2	4898	Real
Car Evaluation	6	4	1728	Int
Image Segmentation	19	7	2100	Int, Real
Shuttle	9	7	54600	Int
Covertype	54	8	581012	Int
Yeast	8	10	1484	Real

TABLE 2.5: Comparison of Cross over Rate

Crossover rate	Average Score of Rules	Iteration
0.01	20.31	28
0.05	20.29	25
0.1	20.24	23
0.2	20.12	23
0.5	19.78	22

TABLE 2.6: Comparison of Mutation Rate

Mutation rate	Average Score of Rules	Iteration
0.01	20.29	28
0.05	20.13	26
0.1	19.98	24
0.2	18.45	20
0.5	14.34	18

2.5.1.1 Comparison of Crossover and Mutation Rate

The main parameters of the proposed method that influences the quality of the extracted rules and iteration time are crossover rate and mutation rate. Therefore, we have added comparisons of several parameter settings of crossover rate and mutation rate using the databases with three attributes and 1000 samples.

Table 2.5 shows the average score of rules and iterations needed to cover all the data when the crossover rate is set at several values. Table 2.5 shows that the increment of the crossover rate slightly reduces the iteration time, and decreases the average score of rules. In this chapter, the crossover rate 0.01 is used to obtain the best average score of rules although the iteration time increases a little. However, the average score of rules does not depend on the crossover rate so much, thus the performance of the proposed method can be stable.

TABLE 2.7: Results of GNP rule extraction with full randomization in six databases

Dataset	Full Random		Partial Random	
	Rule Length	Silhouette	Rule Length	Silhouette
Wine Quality	3.821	0.223	4.233	0.241
Car Evaluation	4.564	0.799	5.1	0.812
Image Segmentation	7.632	0.301	9.333	0.305
Shuttle	3.125	0.328	3.2	0.352
Covertype	5.833	-0.195	12.123	-0.125
Yeast	4.65	0.758	4.75	0.788

Table 2.6 shows the same comparison as Table 2.5 when the mutation rate is set at several values. Table 2.6 shows that the increment of the mutation rate has more effect on the reduction of iteration time and decrease of the average score of rules than the crossover rate. In evolutionary computation, mutation rate is generally set between 0.01 and 0.1, and 0.5 is a too large value. In this sense, if the mutation rate is set between 0.01 and 0.1, the influence of the parameter setting on the average score of rules is not large. From this comparison, we decided to use 0.01 as the mutation rate to obtain the best average score of rules although it slightly increases the iteration time.

2.5.1.2 Comparison of Node Arrangement Methods

The comparison of the results between two node arrangement methods, that is, full randomization and partial randomization, is shown in Table 2.7. The comparison of the results between two node arrangement methods, that is, full randomization and partial randomization, is shown in Table 2. Six databases from UCI are used for the comparison. The performance evaluation is executed to compare the mean rule length and the silhouette value. When the number of attributes of dataset is increased, the number of mean rule length doesn't always shows same increments because of frequent patterns in database doesn't always covers many attributes. However, comparing the mean rule length obtained by full randomization and partial randomization, partial randomization shows better results, i.e., longer frequent feature combination are extracted. Rules are extracted until all the records in the dataset are covered, but the records that have been already covered will not be re-included. The significant difference between full random and partial random is in the average node length where partial random basically shows longer length. By finding frequent item-set (template), partial random basically extracts larger number of longer rules than full random, therefore, silhouette values that obtained by partial random show better results in every dataset. Longer rule length effect to more similarity of multidimensional pattern distribution contained within cluster, which increase silhouette values. From the next section, partial random method is used in the simulations.

TABLE 2.8: Comparison of knapsack rule distribution with dataset "wine quality"

condition #	Number of clusters	Balance	Silhouette
1(original)	2	1599:4898	0.252
2	2	3249:3248	0.238
3	4	800:799:2449:2449	0.342
4	4	1625:1624:1624:1624	0.326
5	6	533:533:533:1633:1633:1632	0.409
6	6	1083:1083:1083:1083:1083:1082	0.393

*Balance : Balance of cluster capacity(the number of data in each cluster)

2.5.2 Knapsack Rule Distribution

In this simulation, it is supposed that the server speck is limited and the storage capacity is different in each server. The clustering results using dataset "wine quality" under six conditions are shown in Table 2.8. "Balance of cluster capacity" shows the proportion of capacity of each site First row (condition 1) shows the original number of clusters and the distribution of data in each cluster. Condition 2 shows the same number of clusters, but the capacity of the clusters is changed to the same. condition 2 shows lower silhouette value than condition 1 because the data distribution is unmatched with the original cluster distribution of the database. Condition 3 shows that each cluster of condition 1 is divided equally into two clusters (totally four clusters). Condition 3 shows better silhouette than condition 1 and 2 because it is easier for larger number of clusters to maintain similarity. Condition 4 shows that the database is divided equally into four clusters. Condition 4 shows lower silhouette than condition 3 because of the same reason to the comparison between condition 1 ad 2. More conditions with six clusters are examined in condition 5 and 6, where condition 5 shows the best results among the six conditions. From the simulation results, it can be found that larger number of clusters and following the original cluster distribution obtain better silhouette.

2.5.3 Comparison with other methods

The five methods for the comparisons with the proposed method are K-means [37], Hierarchical Clustering [38], Fuzzy C means [39], Order-constrained solution in K-means Clustering (OCKM) [40] and K Affinity Propagation [41]. All the methods used in the comparisons are unsupervised clustering methods and use the euclidean distance as a distance metric except hierarchical clustering. The parameter setting of each method is determined as described below :

1. K-means : euclidean distance is used as the distance metric. the value of k is set as the number of classes of each database.

TABLE 2.9: Methods comparison with silhouette evaluation

database	Methods Comparison with Silhouette						
	OCKC	KAP	FCM	K-means	HC	GNP	mean
Wine Quality	0.172	0.182	0.227	0.123	0.224	0.241*	0.195
Car Evaluation	0.795	0.789	0.809	0.801	0.752	0.812*	0.793
Segmentation	0.234	0.265	0.303	0.253	0.296	0.305*	0.276
Shuttle	0.324	0.314	0.398*	0.312	0.354	0.352	0.342
Covertype	-0.214	-0.453	-0.167	-0.254	-0.346	-0.125*	-0.260
Yeast	0.634	0.622	0.779	0.626	0.786	0.788*	0.706
mean	0.324	0.287	0.392	0.310	0.344	0.396*	

TABLE 2.10: Methods comparison with accuracy rate evaluation

database	Methods Comparison with Accuracy Rate						
	OCKC	KAP	FCM	K-means	HC	GNP	mean
Wine Quality	0.642	0.613	0.786	0.771	0.695	0.787*	0.716
Car Evaluation	0.689	0.678	0.699	0.701	0.698	0.701*	0.694
Segmentation	0.678	0.724	0.776	0.725	0.712	0.792*	0.735
Shuttle	0.812	0.787	0.864*	0.839	0.818	0.824	0.824
Covertype	0.675	0.646	0.705	0.676	0.622	0.708*	0.672
Yeast	0.667	0.704	0.812	0.692	0.801	0.856*	0.755
mean	0.694	0.692	0.774	0.734	0.724	0.778*	

2. Hierarchical Clustering : agglomerative is used as the hierarchy strategy and single linkage is used as a clustering method. The clustering procedure finishes when the number of groups reaches the number of classes of each database.

3. Fuzzy C means : Minimum improvement of the fuzzifier m which determines the level of cluster fuzziness is set at 1.0*10-5. The value of k is set as the number of classes of each database.

4. Order-constrained solution in K-means Clustering (OCKM) : euclidean distance is used as the distance metric and recursive dynamic programming strategy is used to improve the clustering quality. The value of k is set as the number of classes of each database.

5. K Affinity Propagation : euclidean distance is used as the distance metric and affinity propagation is used to improve the clustering quality. The value of k is set as the number of classes of each database.

6. Proposed method: The main parameters of the proposed method are crossover rate and mutation rate, and these parameters are determined based on the results in Table 2.5 and 2.6 where several settings of crossover rates and mutation rates

are evaluated in terms of the average score of rules and the iterations needed to cover all the data.

In addition, we have added an accuracy rate as another clustering performance metric whose results for the six data-sets are shown in Table 2.10. Accuracy rate is a common measure used to evaluate how well clustering algorithms perform on a database with a known structure. Accuracy rate shows different result from silhouette depending on the database.

Table 2.9 shows the evaluation result with silhouette and Table 2.10 shows the evaluation result with accuracy rate. Star marks (*) on the side of the results in both tables indicate the best results in each row (database). The proposed method shows the highest average result followed by FCM, HC and K-means, which is shown in the last row of Table 2.9 and 2.10. In both Table 2.9 and 2.10, the proposed method also shows better clustering results in five out of total six databases. The proposed method only loses against FCM and K-means for "shuttle" database. Structure of "shuttle" database, shown in Table 2.4, does not show straight pattern to describe why the proposed method loses against FCM and K-means, but Table 2.10 shows that mean accuracy rate of all the methods (last column of Table 2.10) for "shuttle" database is the highest (0.824), that is, FCM and K-means show better clustering results for the database that is relatively easy to make clusters comparing to other databases.

Here, pay attention to the last column of Table 2.9 and 2.10 showing the mean accuracy rate of all the methods. For example, in Table 2.9, "covertype" database shows very low silhouette value which reaches -0.26, but its average accuracy rate in Table 2.10 is 0.672. In this case, "Covertype" database has the largest number of attributes (54). Silhouette value is very sensitive to the data variation, thus the mean silhouette value of all the methods become lower than other cases (datasets). The similar results are also shown for "wine quality" and "image segmentation" databases. By analyzing such results in Table 2.9, we can find that the large number of attributes tends to decrease the silhouette value because it increases the complexity of attribute combinations, while the large number of classes increases silhouette values because it becomes easier for many clusters to maintain data similarity, in other words, it is difficult for a few clusters to clearly separate many kinds of data fragments.

2.5.3.1 Silhouette values obtained for different numbers of clusters

Generally in clustering problems, the number of clusters is unknown. Therefore, in this simulation, the clustering performance is evaluated when the number of clusters is set at several values.

TABLE 2.11: Comparison of silhouette result using dataset "wine quality" with different clusters setting

Number of clusters	Silhouette						Difference*
	OCKC	KAP	FCM	K-means	HC	GNP	
2 (Original)	0.172	0.182	0.227	0.123	0.224	0.237*	0.051
4	0.204	0.214	0.264	0.243	0.302	0.342*	0.097
6	0.302	0.312	0.324	0.312	0.352	0.404*	0.084
means	0.226	0.236	0.272	0.226	0.293	0.328	0.077

* Difference of silhouette values between GNP and the mean of the conventional methods

TABLE 2.12: Comparison of silhouette result with dataset "shuttle"

Number of clusters	Silhouette						Difference
	OCKC	KAP	FCM	K-means	HC	GNP	
2	0.098	0.132	0.152	0.133	0.197	0.202*	0.06
4	0.233	0.231	0.246	0.233	0.287	0.312*	0.066
7(Original)	0.324	0.314	0.398*	0.312	0.354	0.352	0.012
means	0.218	0.226	0.265	0.226	0.279	0.289*	0.046

* Difference of silhouette values between GNP and the mean of the conventional methods

Table 2.11 and 2.12 show the comparison of silhouette values among GNP and conventional clustering methods using dataset "wine quality" and "shuttle" with several settings of the numbers of clusters, respectively. The proposed method executes clustering under the condition of equal capacity limit for each cluster, i.e., the capacity of each cluster is (the total number of data) / (the number of clusters). The conventional methods only aim to create the predefined number of clusters without capacity limit.

Table 2.11 shows the comparison of silhouette values using "wine quality" dataset with several settings of the number of clusters, i.e., two (original number of clusters), four and six. Table 2.11 shows that the increment of the number of clusters increases the silhouette values of all methods. The proposed method shows the highest mean silhouette value followed by HC, FCM and KAP, which is shown in the last row of Table 2.11. The proposed method shows better results in all of the settings of clusters. The last column shows the difference of silhouette values between the proposed method and the mean of the conventional methods, where the proposed method shows better results in all the three settings.

Table 2.12 shows the comparison of silhouette values using "shuttle" dataset with several settings of the number of clusters, i.e., seven (original number of clusters), four, two. Table 2.12 shows decrement of the number of clusters decreases the silhouette values of all methods. The proposed method shows the highest meanvalue followed by HC, FCM and KAP/K-means (same average results), which is shown in the last row of Table 2.12. The proposed method shows better results except for the original setting (7 cluster)where FCM and HC are better. The last column shows the difference of silhouette values

TABLE 2.13: Comparison of silhouette result of the proposed method without capacity limitation for GNP

database	Methods Comparison with Silhouette						Difference*
	OCKC	KAP	FCM	K-means	HC	GNP	
Wine Quality	0.172	0.182	0.227	0.123	0.224	0.278*	0.092
Car Evaluation	0.795	0.789	0.809	0.801	0.752	0.825*	0.036
Segmentation	0.234	0.265	0.303	0.253	0.296	0.348*	0.078
Shuttle	0.324	0.314	0.398*	0.312	0.354	0.372	0.032
Covertype	-0.214	-0.453	-0.167	-0.254	-0.346	-0.105*	0.182
Yeast	0.634	0.622	0.779	0.626	0.786	0.793*	0.104
mean	0.324	0.287	0.392	0.31	0.344	0.419	0.087

*Difference of silhouette values between GNP and the mean of the conventional methods

between the proposed method and the mean of the conventional methods. Although the proposed method does not show the best value for the original setting among the six methods, the proposed method still shows positive difference to the mean of the conventional methods (0.012).

Both results of table 2.11 and 2.12 shows the proposed method has an ability to keep clustering quality for different numbers of cluster than the other methods. This ability is useful for problems in database management systems such as limited number of storage media (server), which is not always the same as the original number of clusters. The ability of handling capacity limitation is also useful for keeping load balance of each storage media which deals with the same amount of data access.

2.5.3.2 Comparison of silhouette values between the proposed method without capacity limitation and conventional clustering methods

Table 2.13 shows the comparison of silhouette values between the proposed method without capacity limitation and conventional clustering methods. The proposed method shows the highest mean value followed by FCM, HC and K-means, which is shown in the last row of Table 2.13. The proposed method shows better results except for "shuttle" dataset, where FCM is better than GNP (however, in this simulation, GNP is better than HC unlike Table 2.9). The last column shows the difference between GNP and the mean of other methods. Although the proposed method does not show the best result for "shuttle" dataset, the proposed method still shows positive difference from the mean of the conventional methods (0.032).

2.6 Summary

This chapter proposes a novel clustering method combining Genetic Network Programming and Knapsack Problem to handle record clustering with additional storage capacity problem that is compatible with big data with large number of attributes, records and clusters. The proposed method can find good combinations of attributes to create rules for clustering, and also consider the capacity of sites to distribute rules. The clustering performance is evaluated with six datasets downloaded from UCI machine learning repository and the best average results comparing to other five conventional clustering algorithms are achieved. However, the proposed method is less suitable for online processing because of the evolution time to obtain good rules. The proposed method is suitable for an offline processing that requires the optimal results than processing time. Therefore, in Chapter 3, the proposed method is extended to execute online processes of data clustering.

Chapter 3

On-line Rule Updating of Evolutionary Rule Based Clustering

3.1 Chapter Introduction

In this chapter, database cluster method using GNP with on-line rule based clustering is proposed. In the proposed system, an on-line algorithm is utilized to maintain the cluster adaptability against several unbalanced data growth. For example, the unbalanced data growth occurs when different kinds of items (data) comparing to the items stored in the current database begin to be stored as the time goes on (the trend of data is changed). This chapter is organized as follows. Section 3.2 escribes a review of the proposed method, section 3.3 describes the on-line rule updating system, section 3.4 shows the simulation results, and finally section 3.5 is devoted to conclusions.

3.2 Management of Distributed Database

3.2.1 Structure of GNP

Basically, GNP used in this chapter is similar to that used in chapter 2. But in this chapter, a start node to represent the start positions of the node transition is added, while in chapter 2, the start positions are determined by processing nodes. The processing nodes have a different function from chapter 2, that is, the processing nodes represent

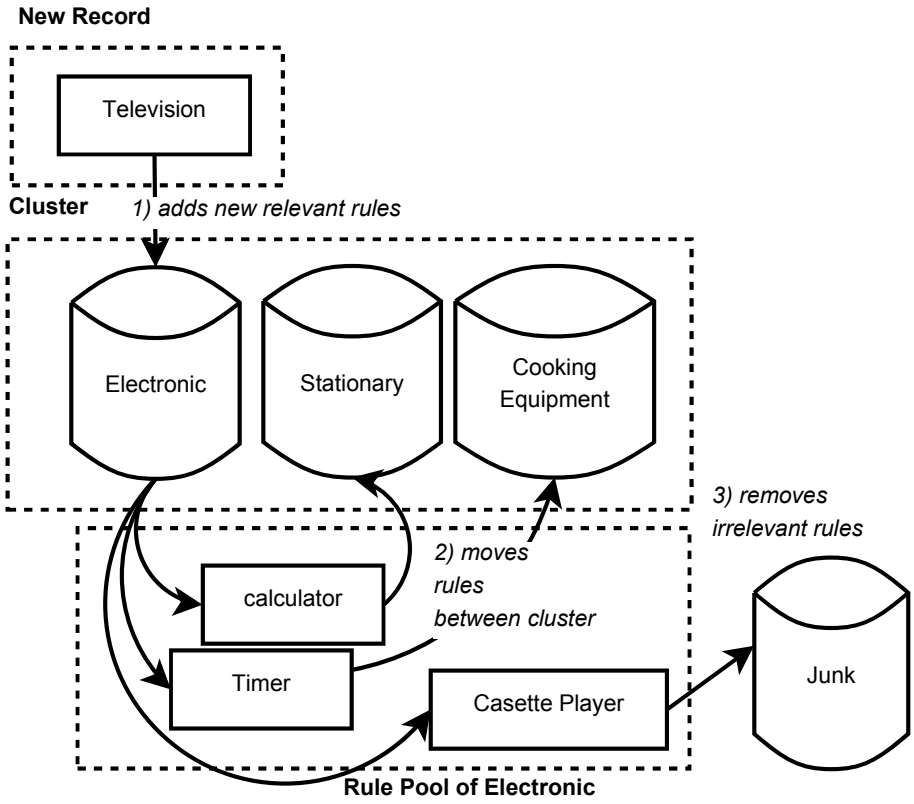

FIGURE 3.1: Cluster Mapping

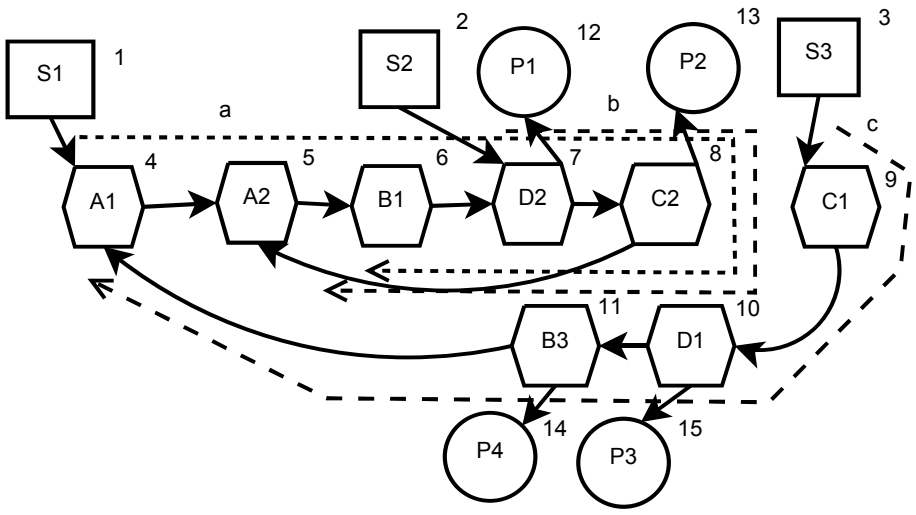

FIGURE 3.2: GNP Implementation on Cluster Optimization.

the cluster numbers to which rules are assigned. Therefore, the structure of GNP in this chapter can be summarized as follows.

A graph structure of GNP consists of three kinds of nodes: start nodes, judgment nodes and processing nodes. Start nodes represent the start positions of the node transition; judgment nodes represent attributes to be examined in a database; and processing nodes

TABLE 3.1: Gene Structure of GNP Corresponding to the Program

i	NT_i	A_i	R_i	C_i
1	1	0	0	4
2	1	0	0	7
3	1	0	0	9
4	2	A	1	5
5	2	A	2	6
6	2	B	1	7
7	2	D	2	8,12
8	2	C	2	5,13
9	2	C	1	10
10	2	D	1	11,15
11	2	B	3	4,14
12	3	1	1	0
13	3	2	1	0
14	3	3	2	0
15	3	4	2	0

TABLE 3.2: Example of Rule Extraction for Cluster Optimization

Start	Rules	Support	Proc	Optimization
1	$A_1 \wedge B_1$	3	-	-
	$A_1 \wedge B_1 \wedge D_2$	1	P_1	Add rule to cluster 1
	$A_1 \wedge B_1 \wedge D_2 \wedge C_2$	1	P_2	Add rule to cluster 2
2	$D_2 \wedge C_2$	2	P_2	Add rule to cluster 2
	$D_2 \wedge C_2 \wedge A_2$	1	-	-
	$D_2 \wedge C_2 \wedge A_2 \wedge B_1$	0	-	-
3	$C_1 \wedge D_1$	2	P_3	Remove rule from cluster 3
	$C_1 \wedge D_1 \wedge B_3$	1	P_4	Remove rule from cluster 4
	$C_1 \wedge D_1 \wedge B_3 \wedge A_1$	1	-	-

Start : Start Node; Rules : Extracted Rules; Proc : Processing Node.

represents the cluster numbers to which rules are assigned. The node preparation for GNP rule extraction contains two phases as the same as chapter 2: node definition and node arrangement. In node arrangement, the template creation and rule extraction combining templates are executed.

3.3 An On-line Rule Updating System

GNP is used to extract rules from a dataset by analyzing all the records. Phenotype and genotype structures of GNP are described in Fig. 3.2 and Table 3.1, respectively. In Fig. 3.2, each node has its own node number i (1–15), and in Table 3.1, the node

information of each node number is described. The program size depends on the number of nodes, which affects the amount of rules created by the program.

1. Start nodes (rectangle) represent the start points of the sequences of judgment nodes which are executed sequentially according to their connections. Multiple placements of start nodes will allow one individual to extract a variety of rules.

2. Judgment nodes (hexagon) represent attributes of the database which are represented by A_i (in Table 3.1) showing an index of attribute such as price, stock, etc., and R_i showing a range index of attribute A_i. For example, $A_i = A$ represents price attribute, and $R_i = 1$ represents value range [0,50] and $R_i = 2$ represents value range [51,80].

3. Processing nodes (round) show the end points of the sequences of judgment nodes and processes the rule updating in a cluster whose cluster number is described as A_i in the processing node.

R_i shows the function of the rule updating, that is, $R_i = 1$ means adding the rule, and $R_i = 2$ means removing the rule. For example, P_1 in Fig. 3.2 (node number $i = 12$ in Table 3.1) processes an addition of extracted rules to cluster number 1. The sequences of nodes starting from each start node (S_1,S_2,S_3) are represented by dotted lines a, b and c. A node sequence flows until support for the next combination of the judgment node (attribute) does not satisfy the threshold. In the node sequence, if the nodes with the attributes that have already appeared in the previous node sequence appear again, the nodes will be skipped.

Candidate rules extracted by the program of Fig. 3.2 are shown in Table 3.2. Table 3.2 shows rule updating is only executed when last judgment node of extracted rules connected to processing nodes. For example, from start node 1, D_1 of second rule $A_1 \wedge B_1 \wedge D_2$ connected to P_1. Its means newly extracted rule $A_1 \wedge B_1 \wedge D_2$ will processed with optimization of P_1 which is "Add rule to cluster 1". Also C_2 of third rule $A_1 \wedge B_1 \wedge D_2 \wedge C_2$ connected to P_2, which have different optimization "Add rule to cluster 2". Placement of the start nodes and processing node also creates variety of optimization to extracted rules.

To create a large number of good rules and optimizations, crossover and mutation are executed.

Crossover: exchange one or more node(s) between parents to make new rules, including placement of the start node(s) and processing node(s).

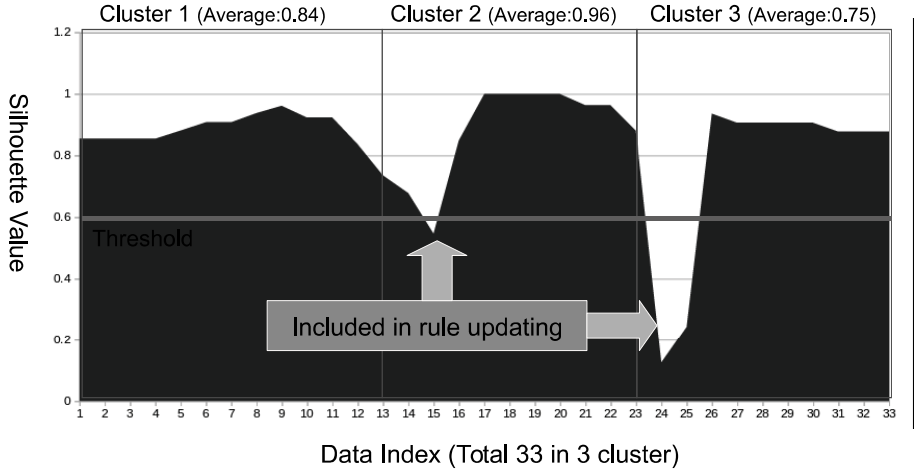

FIGURE 3.3: Example of The Silhouette Values

Mutation: change one or more node(s) to make different combination of nodes, including placement of the start node(s) and function of processing node(s).

Crossover is effective to switch weak nodes (nodes with less improvement to clusters quality) of the parents with strong nodes (nodes with more improvement to clusters quality). Mutation is effective to switch weak nodes of one individual to strong (more functional as improvement to clusters quality) nodes.

When updating rules in each cluster, it is important to find attributes that are not matched with the latest data. Therefore, the attributes to be considered in the rule updating are determined by the following procedure. Fig. 3.3 shows an example of the Silhouette values of each data belonging to each cluster. In Fig. 3.3, threshold is set at 0.5, and only the attributes of data with Silhouette values under 0.5 will be selected for the attributes of the judgment nodes in the rule updating process. The process to select attributes using silhouette for rule updating are described as follows.

1. Using the current clustering result, silhouette values of each data is calculated
2. Find data whose silhouette values are less than the threshold.
3. The attributes contained in the data found in step 2 are selected for rule updating.

The meaning of the above step is that the attributes that are not suitable for the current cluster structure are considered to re-create rules.

In summary, the proposed method consists of two processes:

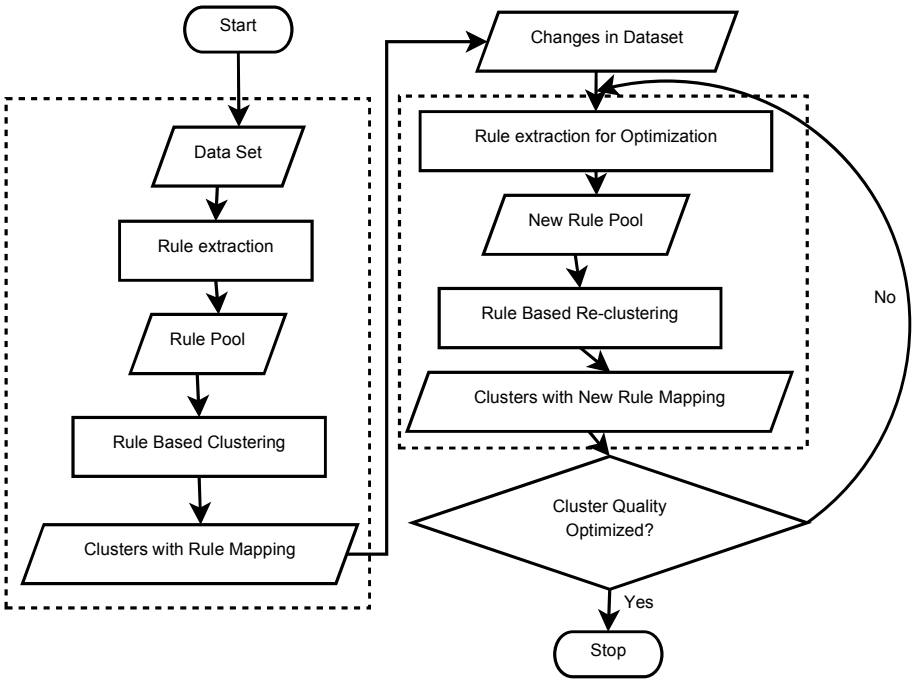

FIGURE 3.4: Flowchart of Proposed Method.

1. Main rule extraction process, which is a standard rule extraction with GNP considering all the attributes in a database. This process is executed to make initial clusters for the initial database;

2. Rule updating process, which is executed for only the attributes that have lower silhouette values than the threshold. This process is repeated until good average value of silhouette (cluster quality) is obtained.

The flowchart of the above processes is shown in Fig. 3.4.

3.4 Simulations

Two kinds of simulations were carried out:

Simulation I: Comparison of silhouette values between different rule updating frequencies;

Simulation II: Comparison of silhouette values and iterations between different setting of thresholds.

TABLE 3.3: Comparison of Simulation Results between Various Rule Updating Frequency

Step	Data	Inc/Dec	Silhouette values			
			-	1000	2000	4000
1	1000 (Default)	-	0.967	0.967	0.967	0.967
2	2000	1000	0.945	0.965*	0.944	0.947
3	3000	1000	0.902	0.923*	0.915*	0.899
4	4000	1000	0.892	0.935*	0.902	0.895
5	5000	1000	0.882	0.912*	0.909*	0.903*
6	6000	1000	0.812	0.902*	0.897	0.887
7	5000	-1000	0.787	0.892*	0.901*	0.821
8	4000	-1000	0.765	0.901*	0.888	0.797
9	3000	-1000	0.723	0.912*	0.892*	0.815*
10	2000	-1000	0.698	0.909*	0.879	0.802
		Average	0.832	0.938	0.923	0.885

3.4.1 Simulation Database

The initial database used in the simulations contains 1000 data with eight attributes. The database is created by randomly determining the attribute values in the fixed ranges of each attribute. For example, attribute 1 has an integer value between 1 and 10, while attribute 2 has a value between 1000 and 2000. To evaluate the adaptability of the proposed method, the number of data will be increased by adding randomly generated data or decreased by deleting data selected randomly.

3.4.2 Comparison of Silhouette Values between Different Rule Updating Frequencies

The first simulation focuses on verifying the cluster adaptability against several unbalanced data growth of the database, where cluster adaptability is evaluated by silhouette values. Unbalanced data growth of database defined as follows :

1. High number of new data patterns that didn't match correctly with stored rules in cluster which data have been added. This case happen commonly because ambiguous definition rules, which usually come from short rules or rules that only define short combination of attributes. In this case new addition of rules are required.

2. Stored rules that no longer have a relevant support for data in cluster because data have been deleted or moved to another cluster (for capacity problem). For this case the rules with low relevancy should deleted or moved to other cluster.

The adaptability is evaluated in terms of the following two points:

1. The number of data in the database is changing as the time goes on. The initial number of data is 1000, then every time step, 1000 new data is added to the database. After the number of data reaches 6000, 1000 data is decreased every time step;

2. The rule updating is executed every after the predefined number of new data are given to the database (the predefined number is called "rule updating frequency"). For example, if the rule updating frequency is 1000, the rule updating is executed every increments or decrements of 1000 data.

The comparisons of the simulation results were carried out between four settings, i.e., the proposed method with rule updating frequency of 1000, 2000 and 4000, and the clustering method of standard GNP without on-line rule updating.

The silhouette values obtained by the four methods are shown in Table 3.3, and its graphical representation is shown in Fig. 3.5. Star marks (*) on the side of silhouette values indicate the times when the rule updating is carried out. In the case of the rule updating frequency of 4000, the increment of silhouette values in step 5 and 9 can be observed, which means that the rule updating is effectively carried out. The best results are obtained by the rule updating frequency of 1000, where silhouette values are stable with relatively high level compared to other frequency parameters. In step 3, although the rule updating is carried out by rule update frequency 1000 and 2000, the silhouette values decreases, because the degree of the data change is relatively large in this step.

3.4.3 Comparison of the Iterations between Different Setting of Thresholds

The second simulation focuses on the comparing of the iteration time and silhouette values between different settings of thresholds for several unbalanced data growth of the dataset. Iteration time in this simulation means the number of individuals generated to cover all the data in the evolution for the rule updating. Lower iteration time is required to minimize hardware resource usage, so on-line processing in this chapter means that the re-organizing the clusters can be executed in less iteration time than the method without on-line rule updating. database used in the simulations has 1000 data with 8 attributes. The adaptability is evaluated in terms of the following two points. 1) The number of data in the database is changed as the time goes on. The initial number of data is 1000, then every time step, 1000 new data is added to the database. 2) Several settings of

TABLE 3.4: Comparison of Silhouette Values and Iteration between Different Setting of Threshold.

Step	Data	Inc	Iteration for each threshold of Silhouette value							
			0.85		0.7		0.5		1	
			Sil	Iter	Sil	Iter	Sil	Iter	Sil	Iter
1	1000 (Default)	-	0.971	153	0.971	153	0.971	153	0.971	153
2	2000	1000	0.945	165	0.935	53	0.923	45	0.912	15
3	3000	1000	0.902	201	0.913	65	0.902	53	0.895	8
4	4000	1000	0.905	265	0.907	89	0.873	75	0.846	11
5	5000	1000	0.902	354	0.913	102	0.851	89	0.802	9
6	6000	1000	0.878	402	0.837	112	0.898	99	0.816	12
Average			0.924	278	0.905	133	0.935	126	0.893	83

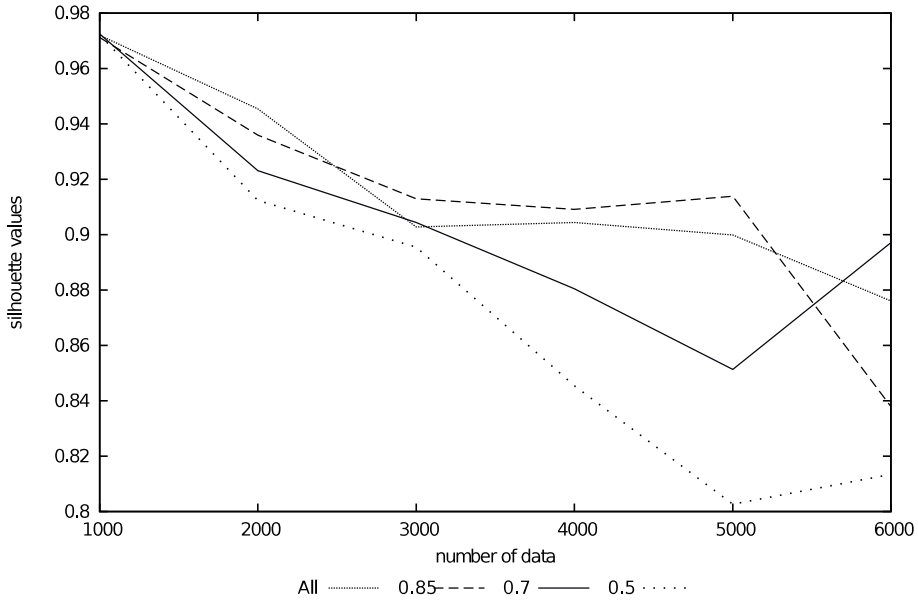

FIGURE 3.5: Comparison of Silhouette Values between Different Rule Updating Frequencies.

thresholds for selecting attributes are analyzed. In this case, the silhouette values of each attribute with the current rules in the cluster is calculated, and if the silhouette values are lower than the minimum value, i.e., a threshold, the attributes showing the lower silhouette values are included in the rule updating process. For example, if the threshold is 0.5, the attributes with silhouette values being lower than 0.5 will be added to the rule updating process. Higher threshold increases the number of attributes to be reanalyzed in the rule updating process, which would increase the iteration times. The comparisons of the simulation results are carried out between four settings of thresholds, i.e., the proposed method with the threshold of 0.5, 0.7, 0.85, and 1.0. The threshold 1.0 means that all attributes will be added to the rule updating process.

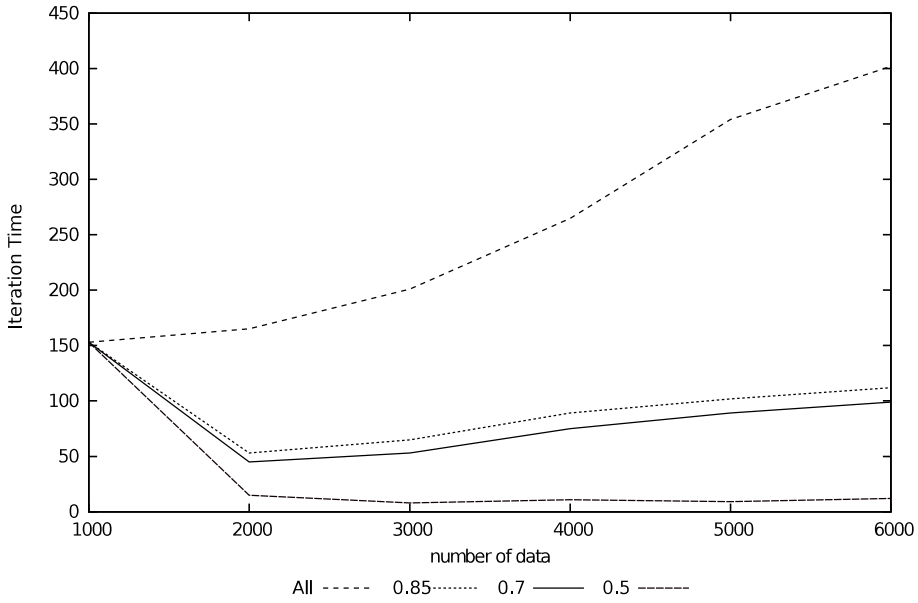

FIGURE 3.6: Comparison of Silhouette Values and Iteration between Different Setting of Threshold.

The silhouette values and iteration times obtained by the four settings of thresholds are shown in Table 5.6, and its graphical representation is shown in Fig. 3.6. The setting of 0.7 shows the best average silhouette that is slightly better than the higher threshold of 1.0 and 0.85. This result shows that the higher thresholds do not always have better re-clustering results. This kind of situations are caused when more attributes are contained in the rule updating process, that is, the possibility to ruin the placement of data, that have been already optimized in the clusters, would increase. Threshold setting of 0.5 has the lowest average silhouette value. This is because the small number of attributes contained in the rule updating process also does not sufficiently optimize the cluster quality. On the other hand, in the comparison of the iteration time, the lowest threshold setting of 0.5 results in the lowest iteration time, and the highest threshold setting of 1.0 results in the highest iteration time. Higher thresholds tend to include more attributes in the rule updating, which will require more iteration times to process many attributes. So when we use the proposed on-line clustering mechanism, the balance between the cluster quality (silhouette values) and the iteration times need to be determined appropriately.

3.5 Summary

This chapter proposed a new on-line rule updating system for maintaining the cluster quality of distributed database with unbalanced data growth. The simulation results of the proposed method showed the better clustering results and iteration time comparing to GNP rule-based clustering without on-line adaptation. Addition of rule optimization

task to GNP structure optimize clustering process without re-process database clustering from beginning, which is suitable as decision support for distributed database management system maintenance. To improve feature representation of database and clustering quality, in next chapter combination with fuzzy object oriented database (FOOD) will be proposed.

Chapter 4

Evolutionary Rule Based Clustering Using Fuzzy Object Oriented Database Models

4.1 Introduction

In the previous chapters, several methods of evolutionary rule based clustering using GNP have been proposed. However, the methods are based on the crisp dataset, that is, describe attributes only with separated ranges, so there are limitation in feature representation problems, for example sharp boundary problem which means no detail of distance between value in same ranges. In this problem minimum and maximum ranges in data separation represented as same quality in rule definition. For example, when judgment node defines price between 100 and 500, minimum (100) and maximum (500) have a same quality as support for this judgment node's definition. To solve such problems, database clustering using GNP with the advantages of fuzzy object oriented database (FOOD) modeling is proposed in this chapter. The main purpose of the proposed method is to provide additional mechanisms to database clustering systems, that is, a data mining algorithm for extracting fuzzy rules, and building clusters based on the extracted fuzzy rules.

This chapter is organized as follows. Section 4.2 describes a review of the proposed framework, section 4.3 describes the details of the proposed framework, section 4.4 shows the simulation results, and finally section 4.5 is devoted to summary of this chapter.

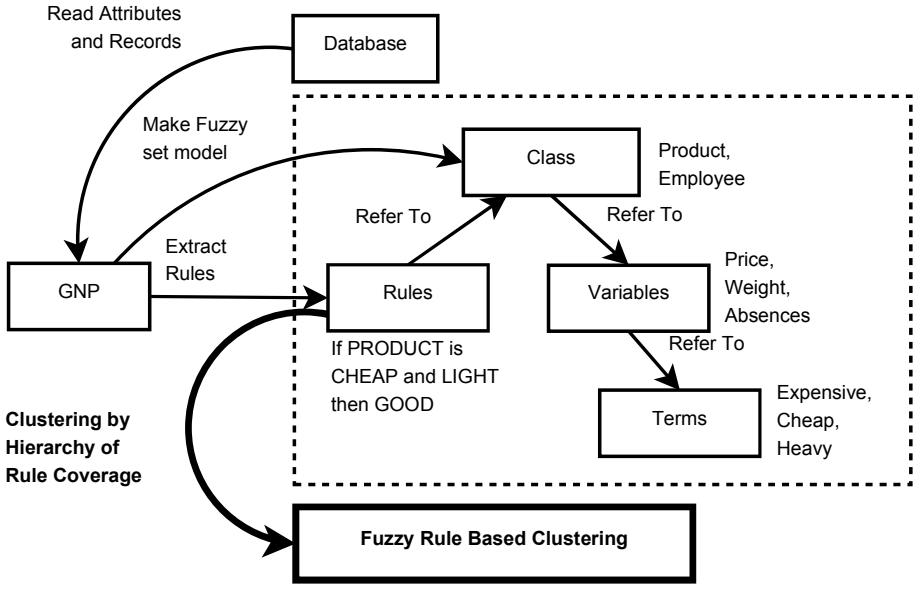

FIGURE 4.1: Fuzzy rule extraction schema

4.2 Review of the Proposed Framework

Adoption of FOOD model to GNP rule extraction process can increase the clustering quality and interpretation of clustering structures. Fig. 5.1 shows the schema of the fuzzy rule extraction. Fig. 5.1 describes the interaction between three components which are database, GNP rule extraction and FOOD. GNP examines a given database and creates a FOOD model containing classes, variables and terms. The detailed process of GNP rule extraction is explained in section 5.3.2 – 5.3.3. After creating the FOOD model, finally, the clustering of the original database is executed by the rule-based clustering, where a similarity measurement on the extracted fuzzy rules has an important role (which is explained in section 5.3.4).

Basically the structure of GNP in this chapter is similar to that in chapter 2. However, the different point of the proposed method in this chapter is that the rules created by GNP represent terms in FOOD model. In addition, the node preparation for GNP rule extraction has been discussed in chapter 2 where there are two phases: node definition and node arrangement, however, with the advantage of FOOD, both phases can be processed simultaneously in the proposed method in this chapter, which results in decreasing the calculation time and enhancing the exploration of the rule extraction.

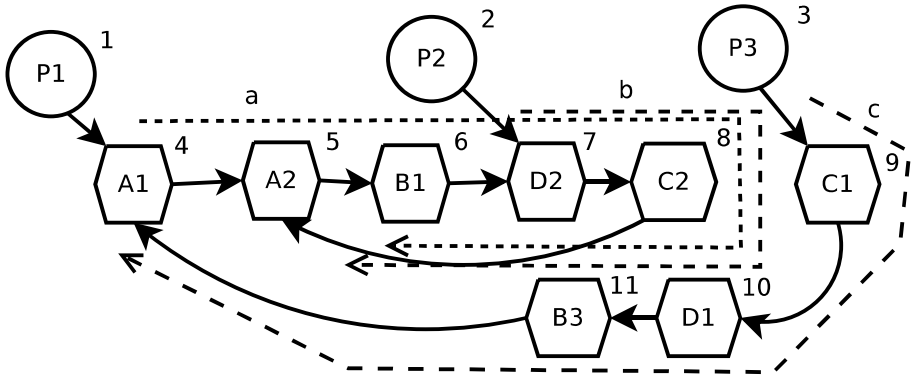

FIGURE 4.2: GNP data mining structure

TABLE 4.1: GNP gene structure of Fig. 4.2

Netwrok			Fuzzy Membership			
i	NT_i	C_i	ATT_i	μ_i	σ_{i1}	σ_{i2}
1	0	4	0	0	0	0
2	0	7	0	0	0	0
3	0	9	0	0	0	0
4	1	5	A	600	35	30
5	1	6	A	800	600	42
6	1	7	B	300	15	53
7	1	8	D	2	1	1
8	1	5	C	520	20	110
9	1	10	C	350	46	26
10	1	11	D	1	0	0

4.3 Detailed algorithm of the Proposed Framework

4.3.1 GNP Rule Extraction with FOOD

$$f(x) = \begin{cases} \exp\left(-\frac{(x-\mu)^2}{2\sigma_1^2}\right) \text{ if } x \leq \mu \\ \exp\left(-\frac{(x-\mu)^2}{2\sigma_2^2}\right) \text{ otherwise} \end{cases} \tag{4.1}$$

GNP is used to extract rules from a database by analyzing all the records. Phenotype and genotype structures of GNP are described in Fig. 4.2 and Table 4.1, respectively. In Fig. 4.2, each node has its own node number i, and in Table 4.1, the node information of each node number is described. The fuzzy membership function used in the proposed method is an asymmetric Gaussian [42] function represented by Eq. 4.1 and graphically in Fig. 4.3.

TABLE 4.2: Fuzzy membership values of sample database

x	Attribute (ATT_i)							
	A_1	A_2	B_1	D_2	C_2	C_1	D_1	B_3
1	0.58	0.58	0	0.09	0.94	0.34	0.62	0.25
2	0.33	0.03	0.71	0.08	0.43	0.51	0.51	0.65
3	0.84	0.81	0.06	0.25	0.23	0.4	0.39	0.58
4	0.92	0.3	0.94	0.63	0.22	0.02	0.03	0.39
5	0.01	0.1	0.13	0.87	0.46	0.03	0.4	0.91
6	0	0.07	0.77	0.86	0.71	0.2	0.47	0.5
7	0.54	0.09	0.04	0.5	0.44	0.46	0.92	0.46
$\overline{x}$	0.46	0.28	0.38	0.47	0.49	0.28	0.48	0.53

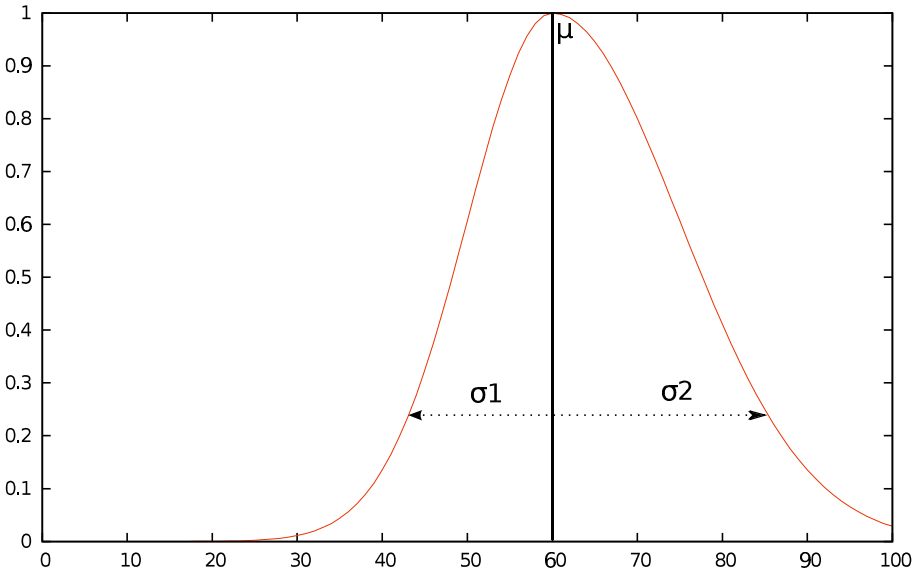

FIGURE 4.3: Asymmetric Gaussian function

where μ is the mean of Gaussian function, σ_1 is a left side standard deviation and σ_2 is the right side standard deviation. The gene structure of GNP in the proposed method is separated into two sections as described below:

1. Network section: used to define the graph structure of GNP.
   - i: node number.
   - NT_i: node type of node i: 0 for processing node and 1 for judgment node
   - C_i: connection of node i, each node i has one connection to the next node whose node number is C_i.
2. Fuzzy membership section: used to define parameters of asymmetric Gaussian fuzzy membership function for each judgment node.

TABLE 4.3: Example of Support calculation in the case of Rule $A_1 \wedge B_1 \wedge D_2 \wedge C_2$

x	A_1	B_1	D_2	C_2	$A_1 \wedge B_1 \wedge D_2 \wedge C_2$
1	0.58	0	0.09	0.94	0
2	0.33	0.71	0.08	0.43	0.08
3	0.84	0.06	0.25	0.23	0.06
4	0.92	0.94	0.63	0.22	0.22
5	0.01	0.13	0.87	0.46	0.01
6	0	0.77	0.86	0.71	0
7	0.54	0.04	0.5	0.44	0.04
Average ($sup(r)$)					0.06

TABLE 4.4: Example of extracted rules and their support and scores

Processing Nodes	Extracted Rules	Support	Score
1	$A_1 \wedge B_1$	0.19	1.19
	$A_1 \wedge B_1 \wedge D_2$	0.12	2.12
	$A_1 \wedge B_1 \wedge D_2 \wedge C_2$	0.06	3.06
2	$D_2 \wedge C_2$	0.32	1.32
	$D_2 \wedge C_2 \wedge A_2$	0.11	2.11
	$D_2 \wedge C_2 \wedge A_2 \wedge B_1$	0.07	3.07
3	$C_1 \wedge D_1$	0.28	1.28
	$C_1 \wedge D_1 \wedge B_3$	0.27	2.27
	$C_1 \wedge D_1 \wedge B_3 \wedge A_1$	0.21	3.21
		Total	19.63

- ATT_i: index of attribute represented by the fuzzy membership function in node i.
- μ_i: mean of Gaussian function in node i.
- σ_{i1}: Left side standard deviation of the Gaussian function in node i.
- σ_{i2}: Right side standard deviation of the Gaussian function in node i.

The program size depends on the number of nodes, which affects the amount of rules created by the program.

1. Processing nodes (circles in Fig. 4.2) represent the start points of the sequences of judgment nodes which are executed sequentially according to their connections. Multiple placements of processing nodes will allow one individual to extract a variety of rules.

2. Judgment nodes (hexagons in Fig. 4.2) represent attributes of the database which are represented by ATT_i (in Table 4.1) showing an index of attribute such as price, stock, etc., and μ_i, σ_{i1} and σ_{i2} showing a Gaussian fuzzy membership function of attribute ATT_i.

For example, in Table 4.1, i=4 and ATT_4=A represent price attribute, μ_4=600 represents the mean of Gaussian function, and σ_{41}=35 and σ_{42}=30 represent the standard deviation of the left side and right side of the Gaussian function, respectively. The sequences of nodes starting from each processing node (P_1,P_2,P_3) are represented by dotted lines a, b and c in Fig. 4.2. A node sequence flows until support for the next combination of the judgment node (attribute) does not satisfy the threshold. For example, the dotted line a extracts the following candidate rules (frequent item sets)

- $A_1 \wedge B_1$
- $A_1 \wedge B_1 \wedge D_2$
- $A_1 \wedge B_1 \wedge D_2 \wedge C_2$.

In the node sequence, if the nodes with the attributes that have been already examined in the previous node sequence appear, the nodes will be skipped. For example, in the node sequence of dotted line a, after visiting node A_1, node A_2 appears. In this case, A_1 and A_2 focus on the same attribute, thus, node A_2 is skipped.

4.3.2 Crossover and Mutation

The aim of crossover and mutation is to change individuals (graph structures) and make various kinds of node combinations so that a large number of rules which can cover whole data spaces are obtained. Crossover in GNP is processed by exchanging one or more gene information of node i between two parent individuals. First, two individuals are randomly selected from 20 individuals with the highest fitness (see Eq. 4.5) in the population (population size is 100). Then, the node numbers for executing crossover are determined randomly, thus it is possible to exchange the gene information between processing node and judgment node. This condition makes the crossover in GNP-FOOD effect more variation of changes.

Mutation in GNP is processed by changing gene values of one or more nodes in a parent individual. After selecting an individual as the same way as the crossover, the mutation changes the values of fuzzy membership sections. For example, if node i is selected for mutation, μ_i, σ_{i1} and σ_{i2} are randomly changed to create totally a new node.

4.3.3 Rule Evaluation and Optimization

Candidate rules extracted by the node sequences in Fig. 4.2 for the sample database in Table 4.2 is shown in Table 4.4. Table 4.2 shows the fuzzy membership values of each

record calculated by the judgment nodes shown in Fig. 4.2, that is, the original database values have been converted to fuzzy membership values. Of course, the values in Table 4.2 are different when GNP individual is different because each individual has its own judgment nodes with different parameters of Gaussian functions. Based on the fuzzy membership values in Table 4.2, each candidate rule can be evaluated by its support value. For example, when a rule $(A_1 \wedge B_1 \wedge D_2 \wedge C_2)$ is extracted by GNP, the support calculation is explained using Table 4.3 which shows the fuzzy membership values of attribute A_1, B_1, D_2 and C_2 of each record. First, pay attention to the row of record $x = 1$, where the membership values of the above four attributes are 0.58, 0, 0.09, and 0.94, respectively. Then, the membership value of rule $r(A_1 \wedge B_1 \wedge D_2 \wedge C_2)$ for record $x(=1)$ is calculated by Eq. 5.6.

$$\begin{aligned} & m_x(r) \\ &= m_1(A_1 \wedge B_1 \wedge D_2 \wedge C_2) \\ &= \min(m_1(A_1), m_1(B_1), m_1(D_2), m_1(C_2)) \\ &= \min(0.58, 0, 0.09, 0.94) \\ &= 0 \end{aligned} \tag{4.2}$$

where $m_x(r)$ shows a membership value of r (an attribute or rule) for record x. Next, as the same way as the first record, the membership values for each record are calculated. Finally, support of rule r is calculated by the average of the membership values of all the records as follows.

$$sup(r) = \frac{1}{N} \sum_{x=1}^{N} m_x(r) \tag{4.3}$$

where N is the number of records.

After obtaining the support value, the score of each rule is calculated by Eq. 4.4 which is based on its support value and rule length.

$$\text{Score of rule } r = \begin{cases} 0 & \text{if } sup(r) = 0 \\ sup(r) + (l(r) - 1) & \text{if } sup(r) > 0 \end{cases} \tag{4.4}$$

where $l(r)$ is the rule length of rule r. The support values and scores of all the extracted rules are shown in Table 1.4. After calculating the support values and scores of all the candidate rules, the fitness of GNP individual is calculated by Eq. 4.5. The fitness calculation is based on the support value, rule length, and the additional score given

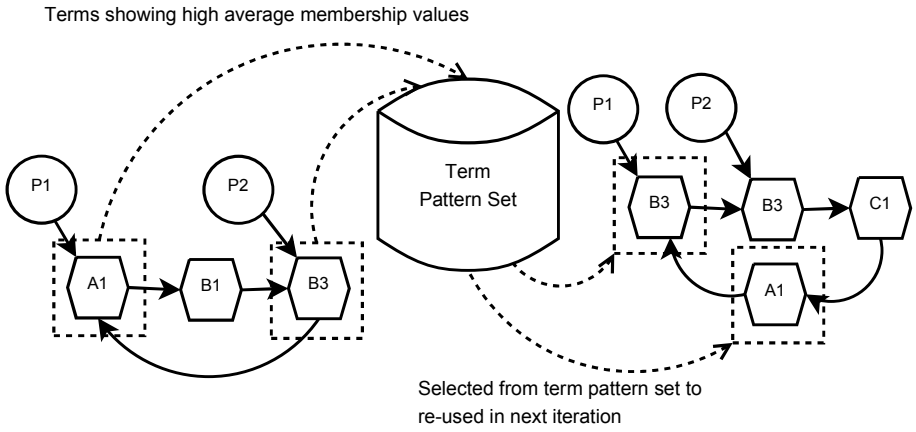

FIGURE 4.4: Fuzzy rule mining of GNP with term pattern set

when a new rule is extracted.

$$\text{Fitness} = \sum_{r \in R} \{sup(r) + (l(r) - 1) + \alpha_{new}(r)\}, \tag{4.5}$$

where $\alpha_{new}(r)$ is an additional value if rule r is newly extracted.

Next, to enhance the efficiency of the evolution, a method that preserves useful terms (fuzzy membership functions) in a term pattern set (Fig. 4.4) is designed. The preserved terms are used in crossover and mutation as partial structures. In the proposed method, terms in the elite individuals and those with high $\overline{x}$ (see Table 4.2) (even if the fitness of the individual is low) are added to the term pattern set. The schema of the term pattern preservation is shown in Fig. 4.4. For example, in Table. 4.2, term B_3 has high $\overline{x}$. Thus, B_3 is added to the term pattern set to be used in the next generation. This mechanism efficiently finds useful terms, therefore, the calculation time can be reduced comparing to the method preparing terms randomly every generation.

4.3.4 Cluster generation

To generate clusters, first, the leader rules with high scores (Eq. 4.4) are selected from the extracted rules and become the centers of each cluster. Then, remaining rules (not selected as the leaders) are put into one of the clusters based on the similarity measurement. Similarity of remaining rules r_1 to the leader rule r_2 is calculated by the following procedure:

1) Before calculating the similarity between rules, the similarity between two attributes, i.e., fuzzy membership functions, is calculated by Eq. 4.6.

$$s\left(a_p^{r_1}, b_q^{r_2}\right) = \frac{\left|\left|A_p^{r_1} \cap B_q^{r_2}\right|\right|}{\left|\left|A_p^{r_1} \cup B_q^{r_2}\right|\right|} = \frac{\int_{-\infty}^{\infty} \min\left\{A_p^{r_1}(x), B_q^{r_2}(x)\right\} dx}{\int_{-\infty}^{\infty} \max\left\{A_p^{r_1}(x), B_q^{r_2}(x)\right\} dx} \tag{4.6}$$

where $a_p^{r_1}$ is a p-th attribute in rule r_1, $b_q^{r_2}$ is a q-th attribute in rule r_2, $A_p^{r_1}$ is a fuzzy membership function of attribute $a_p^{r_1}$, $B_q^{r_2}$ is a fuzzy membership function of attribute $b_q^{r_2}$.

2) The similarity between rule r_1 and rule r_2 is calculated by Eq. 4.7

$$S\left(r_1, r_2\right) = \frac{\sum_{\{p,q\} \in Match} s\left(a_p^{r_1}, b_q^{r_2}\right)}{\max\{N_{ante}(r_1), N_{ante}(r_2)\}} \tag{4.7}$$

where $N_{ante}\left(r\right)\left(r \in \{r_1, r_2\}\right)$ shows the number of attributes in rule r, $Match$ shows a set of suffixes $\{p, q\}$ where $a_p^{r_1} = b_q^{r_2}$ (i.e, matched (same) attributes between rule r_1 and r_2). When the longer rule contains attributes that are not contained in the shorter rule, those attributes are assumed to be matched and the similarity of such attributes (Eq. 4.6) become 1.

After calculating the similarity values between the remaining rules and leader rules of each cluster, the remaining rules are put into the clusters with the highest similarity values.

4.3.5 Data clustering

In the previous subsection, the clusters are generated by rules. Then, a data can be assigned to one of the clusters based on the rules as follows. A data is compared with the rules one by one in ascending order of scores (calculated by Eq. 4.4) until the matched rule is found. Once the matched rule is found, the data will not be compared with other rules, and the data is assigned to the cluster of the matched rule. So, the order of comparison with the leader rules which have the highest scores is the last.

4.4 Simulations

In this section, clustering simulations and the comparisons of the clustering performance between the proposed method and the conventional unsupervised clustering methods are carried out. The conventional methods used for the comparisons are as follows.

TABLE 4.5: Database for simulations

database	Attribute	Class	Record
iris	4	3	150
autompg	7	3	398
heart	13	2	270
wine	13	3	178
bupa	6	2	345
ionosphere	34	2	351
hepatitis	19	2	155
winequality	12	2	6497

TABLE 4.6: Parameter Presets of Proposed Method for Comparison

Preset	Crossover	Mutation
A	0.02	0.05
B	0.1	0.2
C	0.25	0.5

1. Order-constrained solutions in K-means Clustering (OCKC)[40],
2. K-means with Affinity Propagation (KAP)[43],
3. K-means++[44] and
4. Standard K-means[45].
5. Fuzzy C-means Clustering (FCM)[46].

4.4.1 Simulation Datasets

Table 5.5 describes the database names, the number of attributes, classes, and records used in the simulations. The databases were downloaded from University of California at Irvine Machine Learning Repository [47].

The classification accuracy is used as a measurement of the clustering results, which is a common measure evaluating the performance of clustering algorithms on a database with a known structure. The classification accuracy is the percentage of records whose cluster labels are correctly assigned. Higher classification accuracy indicates better clustering results.

TABLE 4.7: Comparison between asymmetric Gaussian with other fuzzy membership functions

Database	Triangle	Bell	Gaussian	Asymmetric Gaussian
Iris	0.872	0.873	0.871	0.872
Hepatitis	0.609	0.611	0.612	0.614
Wine Quality	0.712	0.733	0.745	0.785
Average	0.731	0.739	0.743	0.757

* : Best result

TABLE 4.8: Comparison of Classification Accuracy Between Different Parameter Presets of the Proposed Method

	GNP			GNP FOOD		
	A	B	C	A	B	C
iris	0.889*	0.888	0.888	0.872	0.871	0.868
autompg	0.687	0.685	0.686	0.699*	0.697	0.694
heart	0.602	0.601	0.602	0.725	0.761*	0.712
wine	0.623	0.622	0.624	0.702	0.712*	0.701
bupa	0.574	0.574	0.574	0.573	0.579*	0.571
ionosphere	0.688	0.685	0.687	0.711	0.719	0.723*
hepatitis	0.546	0.547	0.551	0.589	0.612	0.614*
winequality	0.702	0.708	0.701	0.756	0.787*	0.775
Average	0.664	0.664	0.664	0.703	0.717*	0.707

* : Best result

4.4.2 Comparison of accuracy between asymmetric Gaussian with other fuzzy membership functions

Table 6 shows the comparison of accuracy obtained by four types of fuzzy membership functions: Triangle, Bell, Gaussian, Asymmetric Gaussian. The simulation focuses on verifying the cluster quality of the proposed method using several fuzzy membership function. The three databases used in the comparison have the different numbers of attributes, classes, and records as shown in Table 5.5. The result of simulation shown by Table 4.7. The result on database "iris", which is most simple database, shows almost the same results for all the membership functions. The result on database "hepatitis", which has the highest number of attributes (19) and lowest number of data (155) also shows almost the same results. But for database "wine quality", which is the most complicated database with the large number of attributes (12) and data (6497), asymmetric Gaussian shows much better result than the other membership functions. On average, asymmetric Gaussian shows the best result than the other membership functions.

TABLE 4.9: Comparison of the classification accuracy between the proposed method and conventional clustering methods

database	Methods					
	OCKC	KAP	K-means++	K-means	FCM	GNP
iris	0.779	0.825	0.890	0.848	*0.893	0.871
autompg	0.615	0.631	0.631	0.649	0.656	*0.698
heart	0.557	0.610	0.587	0.562	0.593	*0.761
wine	0.625	0.685	0.696	0.588	0.685	*0.712
bupa	0.579	0.578	*0.580	0.577	*0.580	0.579
ionosphere	0.651	0.711	0.639	0.701	0.709	*0.721
hepatitis	0.574	0.556	0.545	0.567	0.594	*0.612
winequality	0.642	0.613	0.779	0.771	0.786	*0.787
Average	0.628	0.651	0.668	0.658	0.687	*0.718

* : Best result

4.4.3 Comparison of Clustering Quality between Different Parameter Presets and with GNP without FOOD

The simulation focuses on verifying the cluster quality of the proposed method using several parameter presets, and the comparison with GNP without FOOD adaptation. The parameter presets on crossover and mutation rates are shown in Table 4.6. These three presets are respectively used in both GNP-FOOD (the proposed method) and GNP without FOOD. The eight databases used in the comparison have the different numbers of attributes, classes, and records as shown in Table 5.5.

The classification accuracy of all the methods for the eight databases are shown in Table 4.8 and its graphical representation is shown in Fig. 4.5. Star marks (*) on the side of classification accuracy in Table 4.8 indicate the best results in each row (database). The preset B of GNP-FOOD shows the highest average accuracy followed by preset C and A of GNP-FOOD, which is shown in the last row of Table 4.8. The preset B of GNP-FOOD also shows better clustering results in four out of total eight databases. All the presets of GNP-FOOD show better classification accuracy, except for "iris" where all the presets of GNP without FOOD show better results than GNP-FOOD. The results are different when the numbers of attributes and records are different, and "iris" is a database with the lowest number of attributes and records. For the databases with larger number of attributes such as "ionosphere" and "hepatitis", preset C of GNP-FOOD shows better result. But for the databases with smaller number of attributes such as "iris" and "autompg", preset A of GNP-FOOD and GNP without FOOD, except database "bupa" for which every method shows the similar result. For the database with large number of records such as "winequality", preset B shows better result. From the above results, we can summarize as follows. For the database with lower number of attributes, large

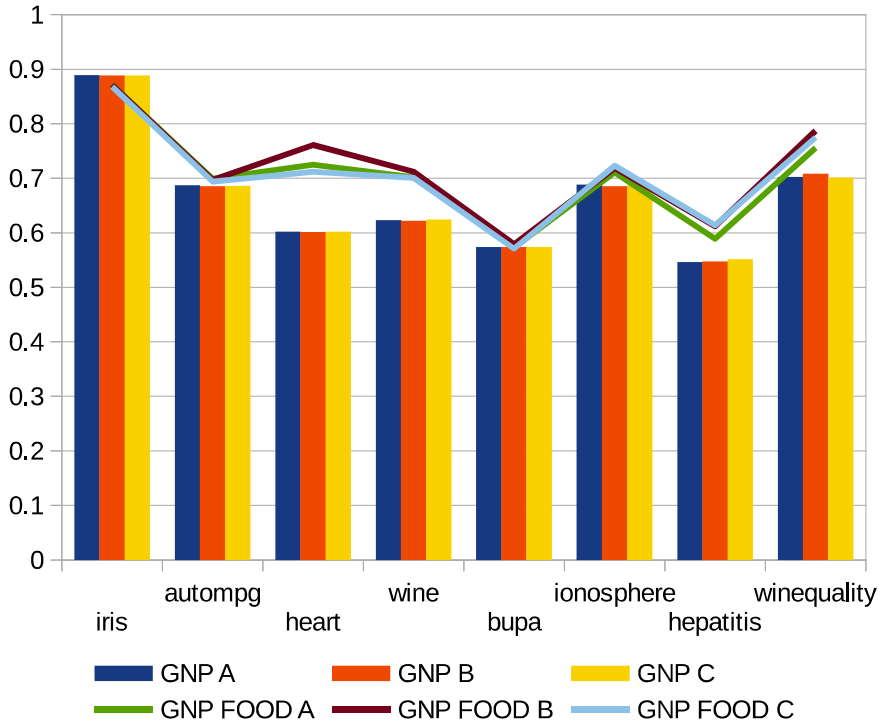

FIGURE 4.5: Comparison of Classification Accuracy Between Different Parameter Presets of Proposed Method

variation of rules is not needed to cover the whole data, thus preset A shows better results. For the database with large number of attributes, preset C shows better results because it generates a variety of rules by changing graph structures largely.

Every preset of GNP without FOOD shows slight difference and the same average accuracy (0.644), which shows that GNP without FOOD is not overly affected by the changes of crossover and mutation rates comparing to GNP-FOOD that shows the variation of results. Such results are caused by the different procedure of crossover and mutation as explained in subsection 4.3.2, where the crossover and mutation of GNP-FOOD have effects on larger number of parameters including fuzzy membership functions comparing to GNP without FOOD which has effects on node functions only.

4.4.4 Comparison of Clustering Quality between the Proposed Method and other Unsupervised Clustering Methods

The simulation focuses on verifying the cluster quality of the proposed method comparing to other unsupervised clustering methods. The databases used in this subsection are the same as the previous simulations.

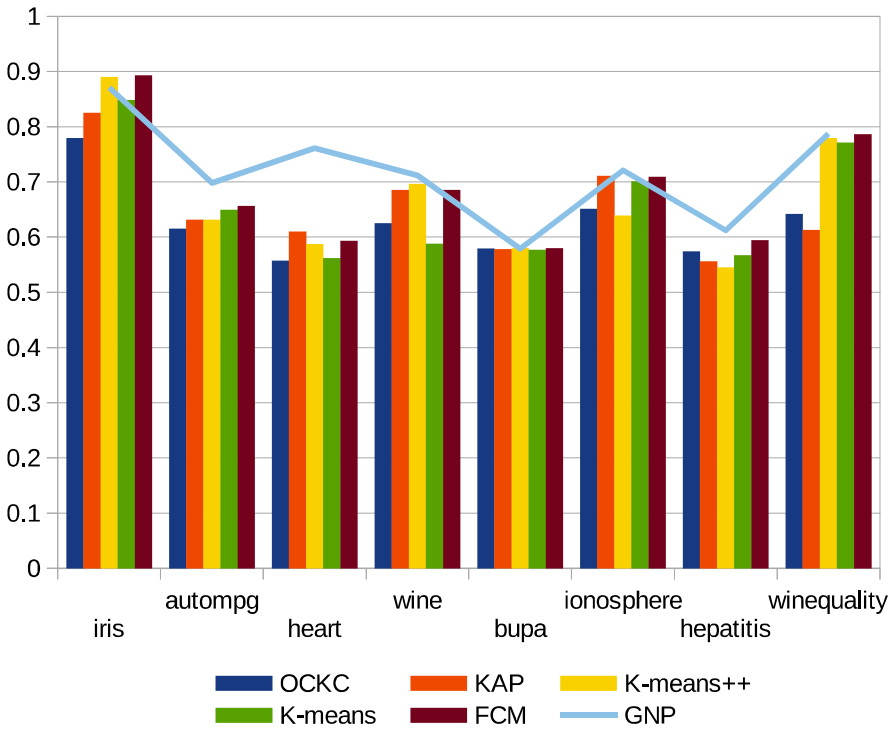

FIGURE 4.6: Comparison of the classification accuracy between the proposed method and conventional clustering methods

The classification accuracy of all the methods for the eight databases are shown in Table 5.6 and its graphical representation is shown in Fig. 5.3. Star marks (*) on the side of classification accuracy in Table 5.6 indicate the best results in each row (database). The proposed method shows the highest average accuracy followed by FCM, K-means++ and K-means, which is shown in the last row of Table 5.6. The proposed method also shows better clustering results in six out of total eight databases. The proposed method loses against FCM and K-means++ for "iris" database which has the smallest number of attributes and records. For database "bupa" with six attributes, every method shows similar results but K-means++ and FCM outperform the proposed method. From the above results, we can see that K-means++ and FCM show better clustering results for the databases with the smaller number of attributes. The proposed method shows better clustering ability for the databases with higher number of attributes and generally shows stable clustering quality for any databases.

4.5 Summary

This chapter proposed a database clustering method using GNP with the advantages of fuzzy object oriented database modeling. The proposed method automatically optimizes

the parameters of fuzzy membership functions and finds good fuzzy rules for making clusters. Addition of fuzzy membership function for node definition increase feature representation, which added detail of distance between each data in same definition with fuzzy membership. More detail feature representation increase accuracy of data clustering process and increase clustering quality. The simulation results of the proposed method showed the better clustering results comparing to the GNP without FOOD and the conventional clustering methods. In next chapter we proposed addition of feature selection to increase clustering ability for high dimensional database.

Chapter 5

Evolutionary Rule Based Clustering with Fuzzy Feature Selection for High Dimensional Database

5.1 Chapter Introduction

Database structures do not always contain same relevancy per attributes for clustering process. The presence of less relevant often decrease clustering accuracy [48]. To solve this problem, feature extraction can be preprocessed with feature selection. Feature selection is the process of identifying the most effective subsets of the original features to use in clustering [3]. Feature selection in which most informative variables are selected for model generation is an important step in data-driven modeling. With feature selection, only attributes with high relevancy are included in or given higher priority in feature extraction.

In this chapter, a database clustering method using GNP with feature selection and representation of fuzzy databases is proposed. The main purpose of this research is to provide additional mechanisms to database clustering, that is, a data mining algorithm for extracting fuzzy rules, and building clusters based on the extracted fuzzy rules for improving cluster quality on high dimensional databases.

This chapter is organized as follows. Section 5.2 describes a review of the proposed framework, section 5.3 describes the details of the proposed framework, section 5.4 shows the simulation results, and finally section 5.5 is devoted to conclusions.

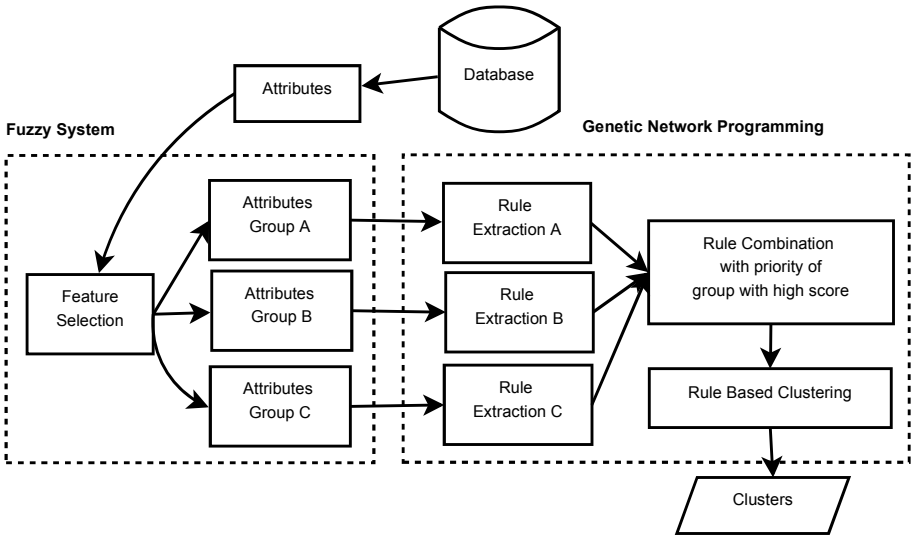

FIGURE 5.1: Fuzzy Feature Selection for GNP Rule Extraction Schema

5.2 Review of the Proposed Framework

5.2.1 Data Matrix and Fuzzy Database

Measurements or events are usually represented as vectors in a multi-dimensional space, where each dimension represents a distinct attribute (variable, measurement) [1, 21]. Set of objects is represented as an m by n matrix, where there are m rows, one for each object, and n columns, one for each attribute. The several types of attributes used in the proposed method are described as follows.

Binary Two values, e.g., true and false.

Discrete A finite number of values, or integers, e.g., counts.

Continuous An effectively infinite number of real values, e.g., weight.

The scales of attributes used in the proposed method are described as follows.

Qualitative

Nominal The values have just different names, e.g., product category, colors.

Ordinal The values reflect an ordering, e.g., high, low.

Quantitative

Interval The values have different unit of measurement, e.g., product price with USD, distances with kilometers.

Ratio The scale has an absolute zero so that ratios are meaningful, e.g., physical quantities such as electrical current, pressure, or temperature on the Kelvin scale.

Fuzzy database is used in the proposed method for feature selection and fuzzy membership mapping is used for the preparation process for GNP rule extraction as described in Fig. 5.1. A database is examined by the fuzzy feature selection method and many fuzzy membership functions optimized for each attribute scale are created. After the feature selection, attributes are grouped by their relevancy, then the rule extraction is performed by GNP group by group independently. The detailed process of GNP rule extraction is explained in section 5.3.2 and 5.3.3. After creating the rules, finally, the clustering of the original database is executed by the rule-based clustering, where a similarity measurement on the extracted fuzzy rules has an important role (which is explained in section 5.3.4).

5.2.2 Structure of GNP

Basically GNP used in this chapter is similar to that used in chapter 4. In this chapter, database is examined by a fuzzy system for feature selection as the replacement of the previous node definition process. Each node contain fuzzy membership of each attribute and grouped by its average of fuzzy membership degree. Node arrangement will be processed separately by each group of attributes. Note that the group of attributes with high number of score will have a priority in making rule extractions.

5.3 Detailed algorithm of the Proposed Framework

5.3.1 Fuzzy Feature Selection and Fuzzy Database Modeling

Fuzzy database modeling in the proposed method can be described as follows. Let a labeled data set be $X = \{X_m | m = 1, 2, ...M\}$, where m is a row number and M is the total number of rows. In a database with multi attributes, each data has its own attribute value set $A = \{a_n | n = 1, 2, ...N\}$, where n is a column number and N s the total number of columns. Then each attribute value x_m^n of set X_m can be represented by a vector :

$$X_m = \{x_m^1, x_m^2, ...x_m^n, ...x_m^N\} \tag{5.1}$$

TABLE 5.1: Example of Fuzzy Database Structure

a_1	a_2	a_3	a_4
f_{11}	f_{21}	f_{31}	f_{41}
f_{12}	f_{22}	f_{32}	f_{42}
f_{13}	f_{23}	f_{33}	f_{43}

TABLE 5.2: Example of Fuzzy Database Feature Selection with Average Membership

a_1	a_2	a_3	a_4
0.82^1	0.01^3	0.21^3	0.74^1
0.46^2	0.64^2	0.87^1	0.36^3
0.24^3	0.12^3	0.48^2	0.24^3

and all the attributes are represented by a set A:

$$A = \{a_1, a_2, ...a_n, ...a_N\} \tag{5.2}$$

Original data are transformed into a fuzzy set by using a membership function set F defined as :

$$F = \begin{bmatrix} f_{11} & \cdots & f_{1i} & \cdots & f_{1I} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ f_{n1} & \ddots & \ddots & \ddots & f_{N1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ f_{N1} & \cdots & f_{Ni} & \cdots & f_{NI} \end{bmatrix} \tag{5.3}$$

where $i(i \in \{1, 2, \dots I\})$ shows index numbers of fuzzy membership function variation for an attribute.

Fuzzy membership function of proposed method using same asymmetric gaussian function that being used by GNP in chapter 4.

Table 5.1 shows an example of fuzzy database structure, which a_n to a_4 shows the number of attributes with each fuzzy membership function that is described as f_{ni}. Table 5.2 shows an example of average fuzzy memberships of database records. Feature selection is executed with these average fuzzy memberships, that is, the nodes with higher fuzzy membership values will be included in the higher priority group. For example, average fuzzy memberships being larger than 0.7 will be added to group one, between 0.4 and 0.7 to group two and remaining values to group three. This grouping will be used as priority in the rule extraction with GNP, which is described in chapter 5.3.4. Groups

of fuzzy features based their relevancy is shown as follows.

$$\begin{aligned} G &= \{g_1, g_2, g_3\} \\ g_1 &= \{f_{11}, f_{32}, f_{41}\} \\ g_2 &= \{f_{12}, f_{22}, f_{33}\} \\ g_3 &= \{f_{13}, f_{21}, f_{23}, f_{31}, f_{42}, f_{43}\} \end{aligned} \tag{5.4}$$

The fuzzy feature selection can be defined by the following steps.

1. Transform a labeled data set (Eq. 5.1), into a fuzzy set F defined by Eq. 5.3. This projection is defined by the asymmetric Gaussian membership functions described by Eq. 4.1.
2. Measure the average membership of fuzzy membership function (Eq. 5.5) f_{ni} of each attribute a_n to data $x_m^n | m \in \{1, 2, \dots M\}$.
3. Make groups of attributes based on the average membership values.

$$\overline{f} = \frac{f_{ni}(x_1^n) + f_{ni}(x_2^n) + \ldots + f_{ni}(x_M^n)}{M} \tag{5.5}$$

5.3.2 GNP Rule Extraction

The rule extraction process is similar to GNP in chapter 4, but the difference is the phenotype and genotype structures of GNP structures. GNP in this chapter does not contain Gaussian function but directly contains fuzzy membership indexes calculated by the previous fuzzy feature selection. The detail of the phenotype and genotype structures of GNP are described in Fig. 5.2 and Table 5.3, respectively. In Fig. 5.2, each node has its own node number i, and in Table 5.3, the node information of each node number is described.

The gene structure of GNP in the proposed method is separated into two sections as described below.

i **:** node number.

NT_i **:** node type of node i: 0 for processing node and 1 for judgment node

C_i **:** connection of node i, each node i has one connection to the next node whose node number is C_i.

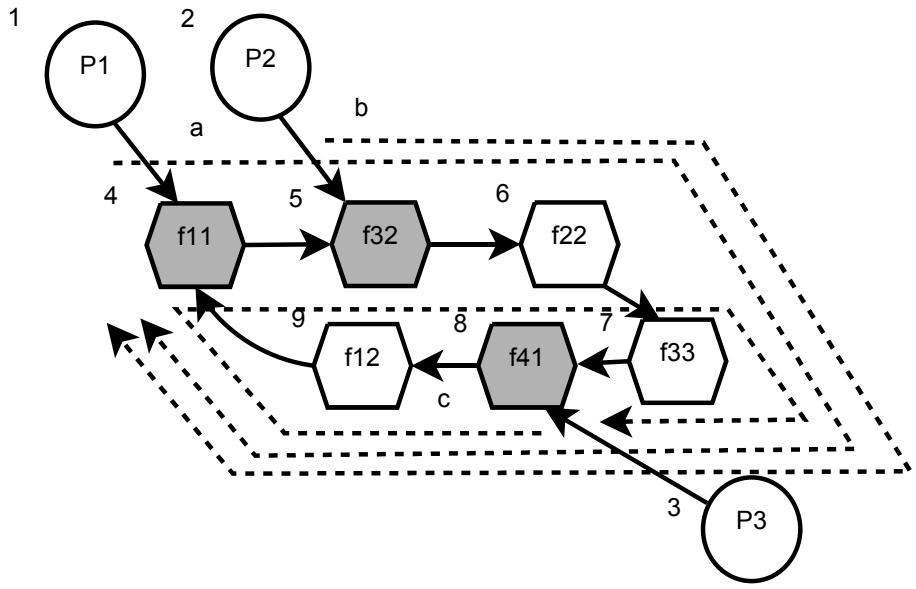

FIGURE 5.2: GNP data mining structure

TABLE 5.3: GNP gene structure of Fig. 5.2

i	NT_i	C_i	f_i
1	0	4	-
2	0	5	-
3	0	8	-
4	1	5	11
5	1	6	32
6	1	7	22
7	1	8	33
8	1	9	41
9	1	4	12

f_i **:** two digit of attribute index and its fuzzy membership function index.

For example, in Table 5.3, i=4 and f_4=12 represent first attribute with its second fuzzy membership function. The sequences of nodes starting from each processing node (P_1,P_2,P_3) are represented by dotted lines a, b and c in Fig. 5.2. A node sequence flows until support for the next combination of the judgment node (fuzzy membership function) does not satisfy the threshold.

The judgment node with gray color in Fig. 5.2 represent fuzzy membership function with high priority and white judgment node for second priority, which are determined by the previous fuzzy feature selection (described in Table 5.1 and 5.2). The processing node which represent the start of node sequence are connected to high priority judgment nodes, so features with high relevancy are always included in rule extraction sequences. In the node sequence, if the nodes with the attributes that have been already examined

TABLE 5.4: Example of extracted rules and their support and scores

Processing Nodes	Extracted Rules	Support	Score
1	$f_{11} \wedge f_{32}$	0.82	1.82
	$f_{11} \wedge f_{31} \wedge f_{22}$	0.64	2.64
	$f_{11} \wedge f_{31} \wedge f_{22} \wedge f_{41}$	0.64	3.64
2	$f_{32} \wedge f_{22}$	0.64	1.64
	$f_{32} \wedge f_{22} \wedge f_{41}$	0.64	2.64
	$f_{32} \wedge f_{22} \wedge f_{41} \wedge f_{12}$	0.46	3.46
3	$f_{41} \wedge f_{12}$	0.46	1.46
	$f_{41} \wedge f_{12} \wedge f_{32}$	0.46	2.46
	$f_{41} \wedge f_{12} \wedge f_{32} \wedge f_{22}$	0.46	3.46
			23.22

in the previous node sequence appear, the nodes will be skipped. For example, in the node sequence of dotted line a, after visiting node f_{11}, node f_{12} appears. In this case, f_{11} and f_{12} focus on the same attribute, thus, node f_{12} is skipped.

In the example of Fig. 5.2, only fuzzy feature group one and two are used as first group of rule extraction. Rule extraction for group three will be performed separately and the extracted rules are stored in different rule pools. This grouping process aims to handle multi-dimensional databases with high number of attributes. High number of attributes and data variation will increase the number of groups of rule extraction but it is still possible to maintain priority of feature relevancy for efficient processing.

5.3.3 Rule Evaluation and Data Clustering

Candidate rules extracted by the node sequences in Fig. 5.2 are shown in Table 5.4. The fuzzy membership values have been calculated by the fuzzy feature selection, so GNP does not repeat the calculation of support of the candidate rules. Based on the average fuzzy membership values in Table 5.2, each candidate rule can be evaluated by its support value. For example, when a rule $(f_{11} \wedge f_{32} \wedge f_{22})$ is extracted by GNP, support is calculated as the minimum average membership of the attribute f_{11}, f_{21} and f_{32}, which is represented by Eq. 5.6.

$$\begin{aligned} & m_x(r) \\ &= m_1(f_{11} \wedge f_{32} \wedge f_{22}) \\ &= \min\left(m_1(f_{11}), m_1(f_{32}), m_1(f_{22})\right) \\ &= \min(0.82, 0.87, 0.64) \\ &= 0.64 \end{aligned} \tag{5.6}$$

where $m_x(r)$ shows a membership value of r (an attribute or rule) for record x. Next, as the same way as the first record, the membership values for each record are calculated.

TABLE 5.5: Dataset for simulations

	Attribute	Classes	Samples	Data Type
Wine Quality	12	2	4898	Real
Image Segmentation	19	7	2100	Int, Real
Covertype	54	8	581012	Int
Yeast	8	10	1484	Real

5.3.4 Cluster generation

Cluster generation in this chapter is the same as in chapter 4, except the leader rules being centers of each cluster are always selected from rule extraction group with the highest priority. Then, the remaining rules are put into one of the clusters in the order of group priority based on the similarity measurement that is used in chapter 4. Note that, the features with low relevancy are also included in the rule extraction, but with low priority. Therefore, the combinations of features with high and low relevancy are still possible. Also, rules that consist of high relevant features and those of low relevant features can be placed in the same cluster as a result of similarity calculation.

5.4 Simulations

In this section, clustering simulations and the comparison of the clustering performance between the proposed method and the conventional unsupervised clustering methods are carried out. The conventional methods used for the comparisons are as follows.

1. Order-constrained solutions in K-means Clustering (OCKC)[40],
2. K-means with Affinity Propagation (KAP)[43],
3. Standard K-means[45] and
4. Fuzzy C-means Clustering (FCM)[46].

5.4.1 Simulation Datasets

Table 5.5 describes the database names, the number of attributes, classes, samples and data type used in the simulations. The databases were downloaded from University of California at Irvine Machine Learning Repository [47].

TABLE 5.6: Comparison of the classification accuracy between the proposed method and conventional clustering methods

database	Methods Comparison				
	OCKC	KAP	FCM	K-means	GNP
Wine Quality	0.642	0.613	0.786	0.771	0.791
Image Segmentation	0.678	0.724	0.776	0.725	0.804
Covertype	0.675	0.646	0.705	0.676	0.774
Yeast	0.667	0.704	0.812	0.692	0.812
mean	0.666	0.672	0.770	0.716	0.795

The classification accuracy is used as a measurement of the clustering results, which is a common measure evaluating the performance of clustering algorithms on a database with a known structure. The classification accuracy is the percentage of records whose cluster labels are correctly assigned. Higher classification accuracy indicates better clustering results.

5.4.2 Comparison of Classification Accuracy Between the Proposed Method and other Unsupervised Clustering Methods

The simulation focuses on verifying the cluster quality of the proposed method comparing to other unsupervised clustering methods. Four databases used in the comparison have the different numbers of attributes, classes, and records as shown in Table 5.5.

The classification accuracy of all the methods on the four databases are shown in Table 5.6 and its graphical representation is shown in Fig. 5.3. The proposed method shows the better accuracy for all the databases and shows highest average accuracy followed by FCM and K-means, which are shown in the last row of Table 5.6. The proposed method shows much better results on database "Covertype" which has the highest number of attributes (54) and samples (581012). The proposed method shows better clustering ability for the databases with higher number of attributes and generally shows stable clustering quality for high-dimensional databases.

5.5 Summary

This chapter proposes a data clustering algorithm using GNP with fuzzy method for feature selection and representation of fuzzy database to optimize cluster generation for high dimensional databases by decreasing the presence of less relevant features. The

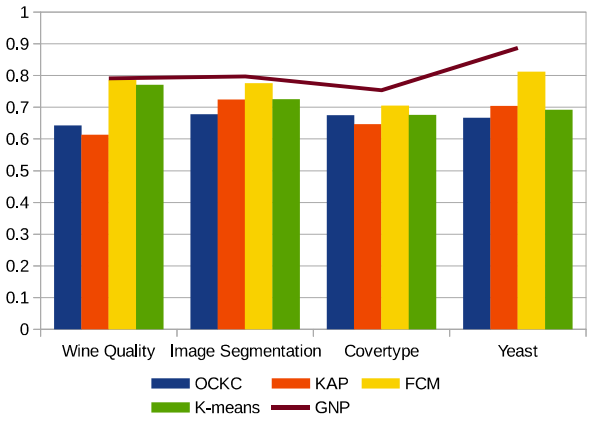

FIGURE 5.3: Comparison of the classification accuracy between the proposed method and conventional clustering methods

simulation results of the proposed method showed the better clustering results comparing to the conventional clustering methods.

Chapter 6

Conclusions

In this thesis, optimization mechanisms of database clustering using evolutionary computation and fuzzy database modeling are proposed, which contains not only the improvement of clustering quality itself, but also the improvement of clustering capability to handle high dimensional databases and the solution for an additional clustering problem of storage capacity limitations.

Chapter 2 proposed the implementation of genetic network programming (GNP) and standard dynamic programming to solve the knapsack problem (KP) as a decision support system for record clustering in distributed databases. Partial random rule extraction method in GNP is also proposed to discover frequent patterns in a database for improving the clustering algorithm, especially for large data problems. The proposed method can find good combinations of attributes to create rules for clustering, and also consider the capacity of sites to distribute rules. The clustering performance is evaluated with six datasets downloaded from UCI machine learning repository and the best average results comparing to other five conventional clustering algorithms are achieved. The expected applications of this chapter is clustering considering load balance. For example, In developing countries or small companies, it may be difficult to prepare many servers with large storage and high performance, for example, each server has different speck. In this case, the proposed method can appropriately distribute data to the servers considering the server speck. For example, a large number of highly accessed data are given to high speck servers, and a small number of low accessed data are given to low speck servers

Chapter 3 proposed a decision support system for database cluster optimization using GNP with on-line rule updating based clustering. On-line algorithm is utilized to maintain the cluster adaptability against several unbalanced data growth. The simulation results of the proposed method showed the better clustering results and iteration time

comparing to GNP rule-based clustering without on-line adaptation. The expected applications of this chapter is decision support for real world applications such as online shopping and search engines, for example, the data and access patterns rapidly change. In this case, the proposed method can maintain rules to adapt to the latest patterns.

Chapter 4 proposed a clustering method using GNP with the advantages of fuzzy object oriented database (FOOD) modeling. The proposed method aimed to provide additional mechanisms to database clustering systems, that is, a data mining algorithm for extracting fuzzy rules, and building clusters based on the extracted fuzzy rules. The simulation results of the proposed method showed the better clustering results comparing to the GNP without FOOD and the conventional clustering methods. The expected applications of this chapter is to realizes user-friendly systems. In travel planning system, for example, when a user gives his/her request such as "South-East Asia, two-persons, seeing beautiful sunset", the most matched plans (clustering results) can be proposed.

Chapter 5 proposed a database clustering method using GNP with the feature selection and representation of fuzzy database as an expansion of the proposed method in chapter 4 that aimed for the improvement of clustering quality on high dimensional databases by decreasing the presence of less relevant or highly correlated features. The simulation results of the proposed method showed the better clustering results comparing to the conventional clustering methods. The expected applications of this chapter is same as chapter 4.

In the future research, it is necessary to execute simulations with real distributed database management systems (DDBMS) with running applications to test the applicability of the proposed method. The proposed method can be also developed as a middle-ware between distributed databases and an application of database fragment allocation management that can access CRUD (Create Read Update Delete) matrix of databases.

Bibliography

[1] Anil K Jain and Richard C Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[2] Edwin Diday and JC Simon. Clustering analysis. In *Digital pattern recognition*, pages 47–94. Springer, 1976.

[3] William HE Day and Herbert Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of classification*, 1(1):7–24, 1984.

[4] Ryszard S Michalski and Robert E Stepp. *Learning from observation: Conceptual clustering*. Springer, 1983.

[5] Richard C Dubes. Cluster analysis and related issues. In *Handbook of pattern recognition & computer vision*, pages 3–32. World Scientific Publishing Co., Inc., 1993.

[6] Sylvain Guinepain and Le Gruenwald. Using cluster computing to support automatic and dynamic database clustering. In *Cluster Computing, 2008 IEEE International Conference on*, pages 394–401. IEEE, 2008.

[7] Barry G Lowden and Athanasios Kitsopanidis. Enhancing database retrieval performance using record clustering. *vectors*, 1(t2):t3, 1993.

[8] Saeed Hassanpour, Martin J. O'Connor, and Amar K. Das. Clustering rule bases using ontology-based similarity measures. *Web Semantics: Science, Services and Agents on the World Wide Web*, 25:1–8, March 2014. ISSN 15708268. doi: 10.1016/j.websem.2014.03.001.

[9] P. R. Bhuyar, A. D. Gawande, and A. B. Deshmukh. Horizontal fragmentation technique in distributed database. *International Journal of Scientific and Research Publications*, 2(5), 2012.

[10] Daniel C. Zilio, Jun Rao, Sam Lightstone, Guy Lohman, Adam Storm, Christian Garcia-Arellano, and Scott Fadden. DB2 design advisor: Integrated automatic physical database design. In *Proc. of the 30th International Conference on Very*

Large Data Bases - Volume 30, VLDB '04, pages 1087–1097. VLDB Endowment, 2004. ISBN 0-12-088469-0.

[11] Sylvain Guinepain and Le Gruenwald. Automatic database clustering using data mining. In *Proc. of the 17th International Workshop on Database and Expert Systems Applications (DEXA'06)*, pages 124–128. IEEE, 2006.

[12] Sylvain Guinepain and Le Gruenwald. Using cluster computing to support automatic and dynamic database clustering. In *Proc. of the 2008 IEEE International Conference on Cluster Computing*, pages 394–401. IEEE, 2008.

[13] Barry G. Lowden and Athanasios Kitsopanidis. Enhancing database retrieval performance using record clustering. *Department of Computer Science, The University of Essex*, 1993.

[14] John H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, 1975.

[15] John R. Koza. *Genetic Programming, on the programming of computers by means of natural selection.* MIT Press, Cambridge, Mass., 1992.

[16] Shingo Mabu, Ci Chen, Nannan Lu, Kaoru Shimada, and Kotaro Hirasawa. An intrusion-detection model based on fuzzy class-association-rule mining using genetic network programming. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 41(1):130–139, 2011.

[17] Kaoru Shimada, Kotaro Hirasawa, and Jinglu Hu. Genetic network programming with acquisition mechanisms of association rules. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 10(1):102–111, 2006.

[18] Deyi Li and Dongbo Liu. A fuzzy prolog database system. 1990.

[19] Sujeet Shenoi and Austin Melton. Proximity relations in the fuzzy relational database model. *Fuzzy sets and systems*, 31(3):285–296, 1989.

[20] Maria Zemankova and Abraham Kandel. Implementing imprecision in information systems. *Information Sciences*, 37(1):107–141, 1985.

[21] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[22] J Galindo, a Urrutia, and Mario Piattini. *Fuzzy databases: Modeling, design, and implementation.* 2006. ISBN 1591403243. doi: 10.4018/978-1-59140-324-1.

[23] J T Cadenas, N Marín, and M A Vila. Context-Aware Fuzzy Databases. *Applied Soft Computing Journal*, 25:215–233, 2014. ISSN 1568-4946. doi: 10.1016/j.asoc.2014.09.020.

[24] K. Gibert, G. Rodríguez-Silva, and I. Rodríguez-Roda. Knowledge discovery with clustering based on rules by states: A water treatment application. *Environmental Modelling & Software*, 25(6):712–723, June 2010. ISSN 13648152. doi: 10.1016/j.envsoft.2009.11.004.

[25] Bilal Sowan, Keshav Dahal, M.a. Hossain, Li Zhang, and Linda Spencer. Fuzzy association rule mining approaches for enhancing prediction performance. *Expert Systems with Applications*, 40(17):6928–6937, 2013. ISSN 09574174. doi: 10.1016/j.eswa.2013.06.025.

[26] Jesús Alcalá-Fdez, Rafael Alcalá, María José Gacto, and Francisco Herrera. Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms. *Fuzzy Sets and Systems*, 160:905–921, 2009. ISSN 01650114. doi: 10.1016/j.fss.2008.05.012.

[27] Gloria Bordogna and Gabriella Pasi. Graph-Based Interaction in a Fuzzy Object Oriented Database. 16:821–841, 2001.

[28] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[29] R. P. Singh. Solving 0–1 knapsack problem using genetic algorithms. In *Proc. of the 2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN)*, pages 591–595. IEEE, 2011.

[30] Katherine Lai. The knapsack problem and fully polynomial time approximation schemes (FPTAS). *18.434: Seminar in Theoretical Computer Science, Prof. M. X. Goemans*, 2006.

[31] JiangFei Zhao, TingLei Huang, Fei Pang, and YuanJie Liu. Genetic algorithm based on greedy strategy in the 0-1 knapsack problem. In *Proc. of the 3rd International Conference on Genetic and Evolutionary Computing (WGEC'09)*, pages 105–107. IEEE, 2009.

[32] Paolo Toth. Dynamic programming algorithms for the zero-one knapsack problem. *Computing*, 25(1):29–45, 1980.

[33] M Tamer Özsu and Patrick Valduriez. *Principles of distributed database systems*. Springer Science & Business Media, 2011.

[34] Hossein Rahimi, Fereshteh-Azadi Parand, and Davoud Riahi. Hierarchical simultaneous vertical fragmentation and allocation using modified bond energy algorithm in distributed databases. *Applied Computing and Informatics*, 2015.

[35] Arkajyoti Saha and Swagatam Das. Automated feature weighting in clustering with separable distances and inner product induced norms–a theoretical generalization. *Pattern Recognition Letters*, 63:50–58, 2015.

[36] John Cuzzola, Jelena Jovanovic, Ebrahim Bagheri, and Dragan Gasevic. Evolutionary fine-tuning of automated semantic annotation systems. *Expert Systems with Applications*, 2015.

[37] Amir Ahmad and Lipika Dey. A k-mean clustering algorithm for mixed numeric and categorical data. *Data & Knowledge Engineering*, 63(2):503–527, 2007.

[38] George Karypis, Eui-Hong Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, 1999.

[39] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2):191–203, 1984.

[40] Douglas Steinley and Lawrence Hubert. Order-constrained solutions in k-means clustering: even better than being globally optimal. *Psychometrika*, 73(4):647–664, 2008.

[41] Xiangliang Zhang, Wei Wang, Kjetil Norvag, and Michele Sebag. K-ap: generating specified k clusters by efficient affinity propagation. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 1187–1192. IEEE, 2010.

[42] Tsuyoshi Kato, Shinichiro Omachi, and Hirotomo Aso. Asymmetric gaussian and its application to pattern recognition. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 405–413. Springer, 2002.

[43] Xianchang Wang, Xiaodong Liu, and Lishi Zhang. A rapid fuzzy rule clustering method based on granular computing. *Applied Soft Computing*, 24:534–542, 2014.

[44] Fouad Khan. An initial seed selection algorithm for k-means clustering of georeferenced data to improve replicability of cluster assignments for mapping application. *Applied Soft Computing*, 12(11):3698–3700, 2012.

[45] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.

[46] James Christian Bezdek. Fuzzy mathematics in pattern classification. 1973.

[47] CL Blake and Christopher J Merz. Uci repository of machine learning databases [http://www. ics. uci. edu/˜ mlearn/mlrepository. html]. irvine, ca: University of california. *Department of Information and Computer Science*, 55, 1998.

[48] Susana M Vieira, João MC Sousa, and Uzay Kaymak. Fuzzy criteria for feature selection. *Fuzzy Sets and Systems*, 189(1):1–18, 2012.