Implementation of Genetic Network Programming and Knapsack Problem for Record Clustering on Distributed Database

Wirarama Wedashwara†, Shingo Mabu, Masanao Obayashi and Takashi Kuremoto

Graduate School of Science and Engineering, Yamaguchi University, Yamaguchi, Japan
(Tel: +81 83-933-5000; E-mail: t001we@yamaguchi-u.ac.jp)

Abstract: This research involves implementation of genetic network programming (GNP) and knapsack problem (KP) to solve record clustering on distributed databases. 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. GNP is used to extract rules from big data by considering characteristics (value ranges) of each attribute in a dataset. 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.

Keywords: Genetic Network Programming, Database Clustering, Knapsack Problem, Record Clustering

1. INTRODUCTION

Distributed database management system (DDBMS) could be a solution for large scale information systems with large amount of data growth and data accesses. DDBMS is recently used in online service websites such as online shops and social networking services that are openly accessible by world wide users.

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). 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 [1].

Fragmentation is a design technique to divide a single relation or class of a database into two or more partitions such that the combination of the partitions provides the original database without any loss of information. This reduces the amount of irrelevant data accessed by the applications of the database, thus reducing the number of disk accesses [2, 3].

DDBMS fragment problems contain the following elements.

Database Structure: How a global relation should be fragmented?
Database Content: What the necessary information for fragmentation and allocation is?
Storage Capacity: How fragments should be allocated to the sites with the limited capacities in the network?
Replication: How many copies of a fragment should be replicated?

Objective of this research is to realize record clustering or horizontal fragmentation. Record clustering allows to find a relation or class to be partitioned into disjoint tuples or instances. Intuition behind horizontal fragmentation is that every site should hold all information that is used to answer queries and the information in the site should be fragmented so the queries of the site run faster.

The objective of the fragmentation is to create fragments of data, distribute them into several sites which have different capacities, and distributed fragments should match with the capacity of each site.

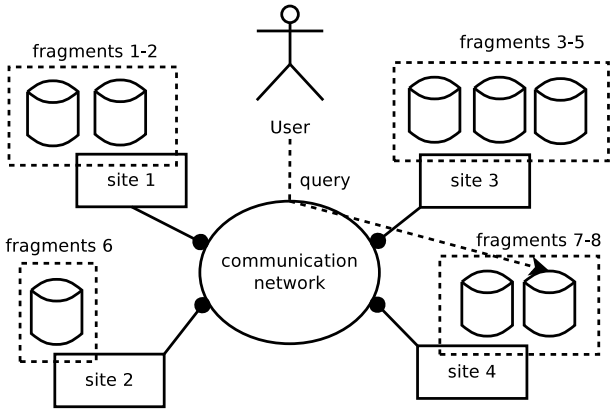

Fig. 1 A distributed database environment

In this paper, a novel method combining genetic network programming (GNP) and knapsack problem for fragment allocation is proposed. Hypothesis of this research are the implementation of GNP for data mining can create effective fragments from complicated dataset and KP can handle the problem of distributing fragments to several sites considering value (similarity of data) and mass (data size) in DDBMS. Therefore, it could be a solution to the fragment allocation and site storage capacity problems.

This paper is organized as follows. Section 2 describes a review of the proposed framework, section 3 describes the detailed algorithm of the proposed framework, section 4 shows the simulation results, and finally section 5 is devoted to conclusions.

2. REVIEW OF THE PROPOSED FRAMEWORK

2.1. Genetic Network Programming

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 pro-

† Wirarama Wedashwara is the presenter of this paper.

Table 1 Example of Frequency Table of Price Attribute

x	f	xf	$(x-\overline{x})$	$(x-\overline{x})^2$	$(x-\overline{x})^2 f$
10	30	300	-65.42	4279.34	128380.21
25	25	625	-50.42	2541.84	63546.01
50	20	1000	-25.42	646.01	12920.14
80	140	11200	4.58	21.01	2940.97
100	65	6500	24.58	604.34	39282.12
150	20	3000	74.58	5562.67	111253.47
Total	300	22625			358322.92

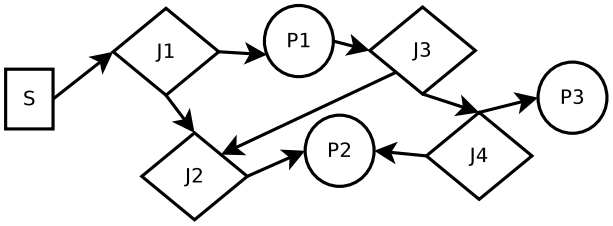

Fig. 2 GNP Basic Implementation

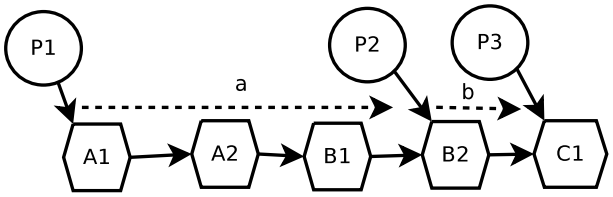

Fig. 3 GNP Implementation on Data Mining

grams derived from the re-usability of nodes in a graph structure.

In GNP, nodes are interpreted as minimum units of the agents judgment and action. Sequences are represented by connecting nodes in a graph structure. GNP does not return to the start node when the actions are completed. The next judgment and action are always influenced by the previous ones. Judgment and processing by a GNP program are performed on the node level.

The basic structure of GNP is illustrated in Fig. 2, with S denoting the start node. There are two kinds of nodes, judgment nodes and processing nodes, with judgment J_p and processing P_q respectively assigned. J_p (p = 1, . . . , n) denotes the p-th judgment, stored in a library for judgment nodes, while P_q (q = 1, . . . , m) denotes the q-th processing, stored in a library for processing nodes [4, 5].

In implementation of data mining, judgement node represents attributes of dataset. Judgement represents a support/suitability of record on dataset to the value of each judgement node. Processing nodes replaced randomly pointed to sequence of judgement nodes and processed sequentially by its index. Only judgement nodes which pointed by processing nodes will be proccesed as rule. For example in Fig. 3 there is a three processing nodes and five judgement nodes. Pointed judgement nodes are $\{A1, B2, C1\}$, so extracted rule represent all record on dataset that supports value of $A1 \cap B2 \cap C1$.

In this research, GNP is used to handle rule extraction from datasets by analyzing the records. Each judgement node of GNP 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 node in GNP. GNP makes rules by evolving combinations of nodes and measures the coverage of the extracted rules. Coverage means that how much data in a database each rule can represent (cover). Rules that pass the coverage threshold will be stored in the rule pool, then in the KP phase, the stored rules are distributed to several sites. The point of this paper is to distribute rules, not the data, which contributes to distributing any data into the sites considering the similarities between rules and data. The detailed explanation of the implementation of GNP in rule extraction is available in section 3.1.

2.2. Knapsack Problem

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.

$$\text{maximize } St = \sum_{i=1}^{n} V_i x_i, \text{ subject to } \sum_{i=1}^{n} w_i x_i \leq W \quad (1)$$

x_i = fragment i; V_i = value (similarity to the leader rule) 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 (knapsack), this optimization can handle the problem of replication [6, 7].

In this research KP is used to handle a distribution of rules extracted by GNP to each site. Rules with high data coverage will be leaders of each site and KP will consider the similarity between the leader rules and remaining rules (which is considered as a value of item (rule) in KP) and coverage of rules (which is considered as weight in KP) should be matched with site capacities. Therefore, the similar rules to a certain leader are mainly put into the same site. Detail explanation of the implementation of KP in the rule distribution is available in section 3.2.

3. COMBINATION OF GNP AND KNAPSACK PROBLEM

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

3.1. GNP Rule Extraction

GNP is used to extract rules from database by analyzing database structure including:

Attributes amount : the number of attributes in a dataset. Each attribute will be spitted 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 dataset.

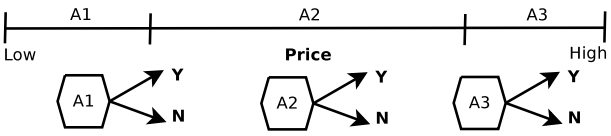

Fig. 4 Node for judging attributes

Table 2 Example of Single Rule with Four Attributes

A1	B3	C2	D1
5-9	30-50	40-79	1-4

Data variation : how much different records are contained in a dataset. If every record in a dataset is different, variation is 100%, if half of record in the dataset is different, variation is 50%, and if every record in a dataset is the same, variation is $1/(\text{the number of data}) \times 100\%$. For example in Table. 1 there are six data variation in total 300 data, so variation will be $(6/300) \times 100 = 2\%$.
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.

3.1.1. Node Definition

Main purpose of node definition is to define nodes that will be combined to create rules. First step is to find the minimum and maximum value of each attribute. For example, the minimum value of "price" attribute is 10 and the maximum value is 150 in the dataset with 300 records. Then, a frequency table is created per attribute as shown in Table 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. Mean $(\overline{xf})$ and standard deviation S are calculated by Eq. 2, where $(\overline{xf})$ is used here and S will be used in section 3.1.3. as spread measurement of attributes.

$$\begin{aligned} \overline{xf} &= \frac{\sum xf}{\sum f} = 75.42 \\ S &= \sqrt{\frac{\sum (x-\overline{x})^2 f}{\sum f}} = 34.56 \end{aligned} \tag{2}$$

To define nodes from Table 1, data should be divided equally based on data amount. For example, three nodes could be created by dividing value range into three ranges considering the occurrence frequency as shown in Fig. 4. In this example, three ranges are: $x = \{10, 25, 50\}$ (75 data), $x = \{80\}$ (140 data) and $x = \{100, 150\}$ (85 data). First node and third node contain more than one prices 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 frequency to become a node. Through the measurement, second node can created from single record ($x = \{80\}$ with 140 data).

Fitness function of extracted rule measured by its coverage. Coverage of rules measured by compare each attribute range to record on data set. Table. 2 shows example of extracted rule with four attributes and Table. 3 shows example of dataset that used to be covered. Coverage threshold is used to determine passing status of records as covered or not. In this example, if coverage threshold is 3/4, data 1 and 2 are covered. Value of coverage threshold mostly increase the coverage rate of extracted rules but decrease the cluster quality.

Table 3 Example of Dataset to be Covered

A	B	C	D	match
8	12	56	1	3/4
6	45	45	2	4/4
2	23	43	2	2/4

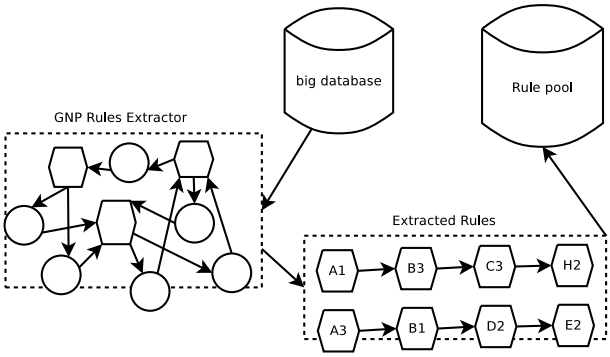

Fig. 5 GNP rule extraction

3.1.2. 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 3.1.1 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. 5). How to extract rules from the graph structures is described in [5] in detail.

After rules are extracted, GNP will measure the amount of coverage archived by the rules. In this research, coverage of rule *r* means the number of records that match (covered by) the rule *r*. If the coverage exceeds the coverage threshold, such rules will be added to rule pool, otherwise, the rules will be discarded. Rules with high coverage will be defined as elite rules and will be the leaders of each cluster (site) in KP process.

Rule extraction process will continue until :

- All data in the database are covered, and
- Minimum amount of coverage of all rules are archived.

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 each 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.

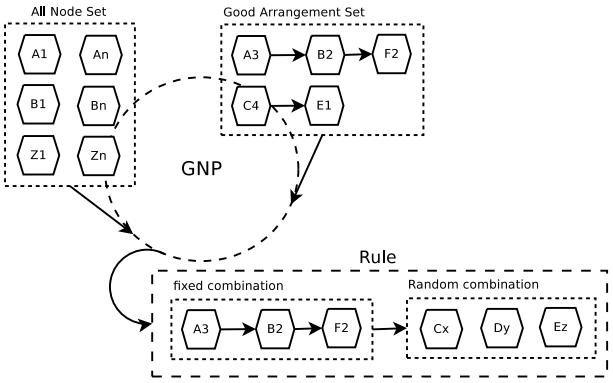

Fig. 6 Node Arrangement Optimization in GNP

3.1.3. Nodes Arrangement : Partial Random

As described in section 3.1.1, every attribute may have unbalanced frequency of data. Randomly combining different attributes results in decreasing the coverage. For example, in a database, there are 500 products that have a price around 50-100[USD], and 200 products in stationary category. However, suppose there are only five products that are matched with both conditions. In this case, "price[USD] $\in$ [50,100]" and "category = stationary" are a bad combined condition. On the other hand, if there is 100 products that are matched with condition of price[USD] $\in$ [1,10] and category = stationary, combining these conditions would make good rules.

To find a good arrangement, first, attributes with the low standard deviation (Eq. 2) are selected. Then, one or more nodes with the highest frequency f of the selected attribute will be randomly connected as short rules (templates). If the combinations of these templates and other nodes produce good rules with high coverage, the templates are stored in the good arrangement set (Fig. 6). For example, in Table. 1, second node : x = [80] (140 data) will be used in node arrangement as a high frequency node.

When initializing GNP structures, node arrangement with partial random method does not randomly prepare the nodes of attributes, but randomly prepare only a part of the nodes, and the remaining nodes are selected from good arrangement (combination) set to find much better combination of nodes (rules). As a result, GNP can extract rules with high coverage.

The objective of this method is to find general characteristics of data that is difficult to be found by full randomization. For example, most of stationary products have a price under 10[USD] and weight less than 100 grams. That would be general characteristics of stationary and other attributes such as size, color etc. may be varied but generally does not dominate the characteristics of stationary.

Table. 4 shows example of nodes with high amount of coverage. *Attribute* shown nodes name, alphabet describe attribute index and numeric describe range index. In example A1 and A2 are same attribute with different range which shown by different minimum*min* and maximum*Max* value. *S* shown standard deviation of each attribute. In example A1 and A2 have a same standard deviation because they are on same attributes.

Table 4 Example of Attribute to be Used on Partial Random

Attribute	Min	Max	S	Coverage
A2	149	246	36	28
B3	660	1626	48	75
A1	42	102	36	68
D3	1592	2340	47	30

Table 5 Example of Combination of Partial Random

$x \cap y$	A2	B3	A1	D3
A2	-	18	-	22
B3	18	-	15	23
A1	-	15	-	21
D3	22	23	21	-

Table 6 Example of Combination of Partial Random with Remaining Attributes

GBS	RN1	RN2	Coverage
$A2 \cap D3$	B1	C2	0
$A2 \cap D3$	B3	C2	10
$A1 \cap D3$	B3	C2	14
$B3 \cap D3$	A1	C2	12

In partial random method attributes with high amount of coverage will combined each other like shown in Table. 5 and combination with high amount of coverage will be added in fixed combination set. As shown in Table. 6 three combination that have a coverage more than 20 is selected. To make a complete rule, fixed combination will randomized with remaining attributes that not included in fixed combination. In Table. 6, when $A2 \cap D3$ is selected, remaining attributes that will be randomized are B and C. In some case coverage can be zero although have used fixed combination of nodes with high coverage, because in dataset with random record there is not always contain every combination of value.

3.2. Rule Distribution with Knapsack Problem

After all the records in a dataset are covered by rules extracted by GNP, KP is used to distribute rules to several sites. Rules with high coverage (elite) will be the leaders of each site, then KP will consider the similarity of the remaining rules to the leader rules (value) and coverage of the rules (weight) in order to distribute the remaining rules to the sites. This process is shown in Fig. 7

Euclidean distance (Eq. 3) [8] is used to measure distance between remaining rule (P) and leader rule (Q) in each site.

$$\begin{aligned} d(P,Q) &= \sqrt{\sum_i (p_i - q_i)^2} \\ &= \sqrt{(p_1 - q_1)^2 + ... + (p_n - q_n)^2}, \end{aligned} \tag{3}$$

where,

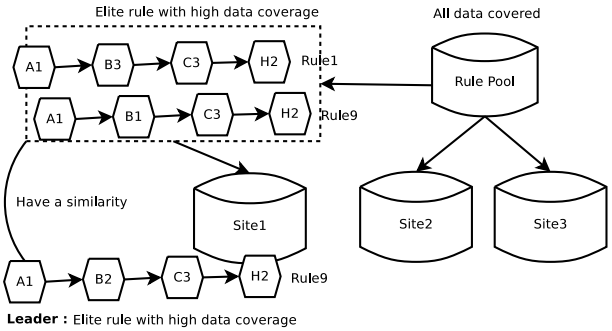

Fig. 7 Rule Distribution with Knapsack Problem

Table 7 Nodes of Each Attribute with its Range and Distance

	A	B	C	D
1	[5,9];5	[5,19];15	[10,39];30	[1,4];4
2	[10,13];4	[20,29];10	[40,79];40	[5,8];4
3	[14,20];7	[30,50];21	[80,100];21	[9,10];2
	16	46	91	10

Table 8 Remained Rules

Rule	A	B	C	D	dist	ED(%)
1	A1	B2	C1	D2	2	54.71
2	A2	B3	C2	D1	1	31.25
3	A1	B1	C2	D1	1	32.61

p_i : value of attribute i contained in remaining rule P,
q_i : value of attribute i contained in leader rule Q,
n : number of attributes.

The values of each attribute are normalized to be the same standard in the distance calculation. For example Table. 7 shows nodes of each attribute with its ranges and distance. Last row shows total distance of each attribute. Table. 8 shows remained rules to be compared with leader rule which shown in 2. *dist* shows number of attributes that have a distance from leader rule which if node is in same range its will counted just as zero. Calculation of euclidean distances shown on 4. If distance threshold is 60% but site capacity is just two records, only two rules with lowest euclidean distance will be added to site which in example are rule two and three. Rule one will be added to another site which possibly meet with leader rule that have a lower distance than current site.

$$
\begin{aligned}
Rule1 &: ((B3 - B2)/B) + ((C2 - C1)/C) \\
&= 10/46 + 30/91 \\
&= 21.74\% + 32.97\% = 54.71\% \\
&= 54.71\% \\
Rule2 &: ((A2 - A1)/A) \\
&= 5/16 \\
&= 31.25\% \\
Rule3 &: ((B3 - B1)/B) \\
&= 15/46 \\
&= 32.61\% \\
d(P,Q) &= \sqrt{54.71^2 + 31.25^2 + 32.61^2} \\
&= 70.94
\end{aligned}
\tag{4}
$$

In some cases, there is no large distance or almost the same distance between clusters. Such cases happen depending on the site capacities and distance between leader rules. If similar data cannot be included in one site due to the limitation of capacity, the remaining data will be distributed in a new site or another site that still has a space.

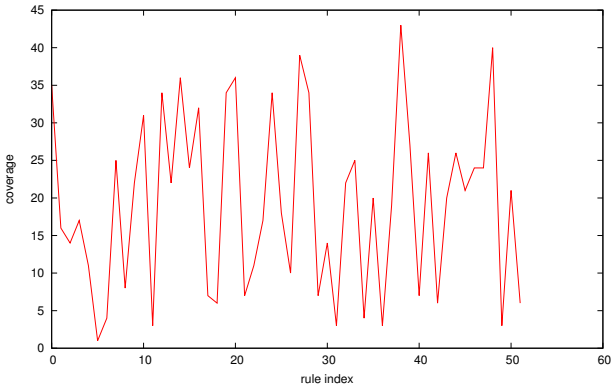

Fig. 8 Coverage of the simple dataset

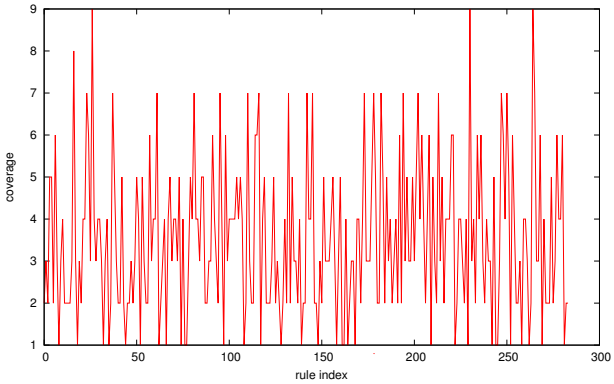

Fig. 9 Coverage of the complicated dataset

4. 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 k-means and hierarchical clustering.

4.1. GNP Rule Extraction

The coverage of rules and the number of extracted rules depend on the complexity of a dataset. Examples are shown in Figs. 8 and 9. The horizontal axis shows the rule numbers, and the vertical axis shows the coverage of each rule. In the simple dataset with less data variation (Fig. 8), maximum value of coverage reaches around 40. But some rules cover only one–two records. Such situation occurs when some data have unique values and are difficult to be covered by general rules. Fig. 7 shows the coverage of each rule when the dataset is complicated, where the maximum coverage is nine. Therefore, more rules are necessary to cover all the records in the complicated dataset.

Figs. 10 and 11 show the number of extracted rules v.s. coverage. When the dataset is simple as Fig. 10, 250 extracted rules can cover all the 1,000 data in a dataset,

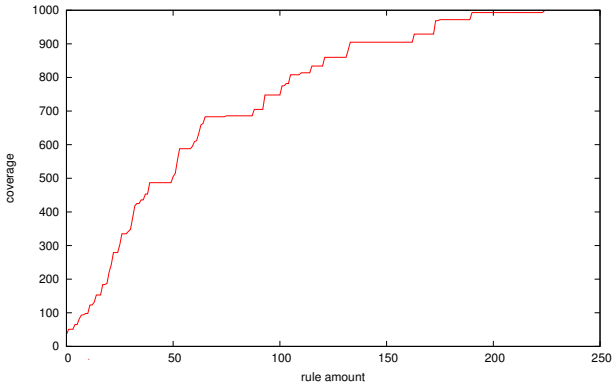

Fig. 10 Relation between the number of extracted rules and coverage in the simple dataset

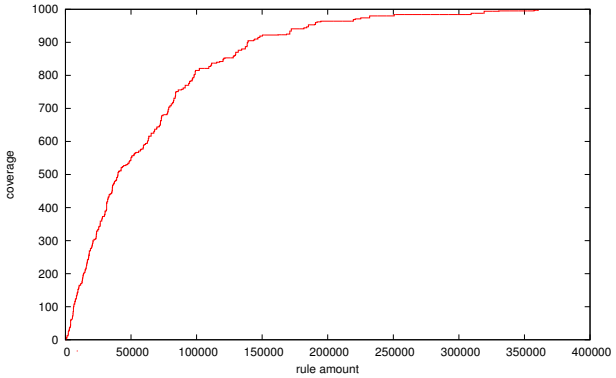

Fig. 11 Relation between the number of extracted rules and coverage in the complicated dataset

i.e., 100% coverage. However, when the dataset is complicated as Fig. 11, around 400,000 rules are needed to cover all data. The coverage increases rapidly at the beginning of rule extraction but more slowly towards the end. It means that the records that have been already covered by previously extracted rules are not recovered by another rule. Therefore, the remaining data gradually become difficult to be covered.

Result comparison between two node arrangement methods, that is, full randomization and partial randomization, are shown in Tables 9. Six datasets are used for comparison, where the number of data and data variation are different. The performance evaluation is executed based on the iterations needed to cover all the data, the number of rules covering the data, and coverage value. Here, iteration means one time rule extraction. When the number of attributes and data variation are increased, the number of iterations needed to cover all the data tends to be increased. However, comparing the iteration needed by full randomization and partial randomization, partial randomization shows better results, i.e., less iteration are needed. The number of rules obtained by both methods are not different so much, but coverage obtained by partial randomization show better result because it can create good combinations of attributes (nodes).

4.2. Knapsack Rule Distribution

Here, silhouette value (Eq. 5) is used to evaluate the clustering results. Silhouette provides a succinct graphical representation of how well each object lies within its cluster [9].

Table 10 Result of Knapsack Problem (Silhouette values)

k	Size of Each Site	Average	Max	Min
4	1:1:1:1	0.98	0.99	0.95
4	1:4:2:1	0.95	0.99	0.87
6	1:1:1:1:1:1	0.96	0.99	0.91
6	1:5:2:6:3:2	0.94	0.99	0.86
8	1:1:1:1:1:1:1:1	0.94	0.98	0.88
8	4:2:4:6:4:2:7:5	0.89	0.98	0.79

$$s = \frac{b-a}{\max\{a,b\}} = \begin{cases} 1-a/b, & \text{if } a < b \\ 0, & \text{if } a = b \\ b/a - 1, & \text{if } a > b \end{cases} \quad (5)$$

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: mean distance between a sample and all other points in the same cluster.
b: mean distance between a sample and all other points in the second nearest cluster.

The results of rule distribution are shown in Table. 10. k is the number of clusters (sites). "Size of Each Site" shows the capacity of each site. For example, 1:1:1:1 means all the four sites have a same size, and 1:4:2:1 means that the second site (size four) is four time larger capacity than the first site (size one). "Average, Max and Min" show the data on Silhouette values obtained by generated clusters.

4.3. Comparison with K-means and Hierarchical Clustering

Clustering methods that will be used for the comparison are k-means and hierarchical clustering because both of the methods also use euclidean distance. K-means is based on a centroid concept for cluster separation and the centroids also represent each cluster. Hierarchical clustering is selected for comparison because similar concept to the proposed method using leader rules is used to distribute rules. Although both methods can set the number of clusters to be created, they does not have a function to measure cluster capacity as KP. Thus, cluster capacity problem is not discussed in this comparison. The features of attributes contained in the dataset for comparison is shown in Table 11. In this simulation, eight datasets are prepared, where the values of each record are determined randomly in the ranges of each attribute, and the number of records and record variation are different.

Table 12 shows more information on the eight datasets (including simple and complicated datasets) and Silhouette values obtained by the proposed method (GNP-KP), K-means and hierarchical clustering (HC). Fig. 12 graphically shows the silhouette values. When the number

Table 9 Results of GNP rule extraction with full randomization in six datasets

			Full Random			Partial Random		
Attribute	Data	Variation	Iteration	Rule	Coverage	Iteration	Rule	Coverage
3	1000	50.00%	410	62	15.87	93	61	16.84
3	10000	20.00%	431	29	344.82	121	27	367.24
8	1000	30.00%	458192	39	26.64	387293	39	26.64
8	10000	60.00%	512239	70	145.89	435729	69	153.56
15	1000	70.00%	98245k	76	13.15	98189k	72	14.67
15	10000	10.00%	2678k	15	62.57	2537k	12	67.38

Table 11 Value ranges of attributes in the dataset

Attribute	Min Value	Max Value
1	100	500
2	1000	2000
3	700	1500
4	1	5
5	15	95
6	10000	30000
7	5000	50000
8	1	10

Table 12 Data on Silhouette values in the eight cases

k	attr	data	var	K-mean	HC	GNP-KP
2	2	1000	30	0.981	0.873	0.982
8	2	5000	50	0.963	0.871	0.972
2	4	1000	30	0.885	0.790	0.927
8	4	5000	70	0.793	0.648	0.915
6	8	3000	30	0.476	0.591	0.784
8	8	10000	50	0.402	0.579	0.730
3	10	1000	30	0.345	0.559	0.666
8	10	5000	70	0.302	0.500	0.656

k: the number of clusters, attr: the number of attributes, var: variation[%]

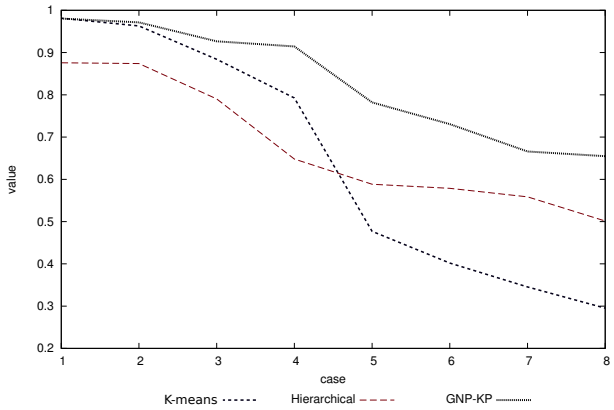

Fig. 12 Comparison of Silhouette values in eight cases

Table 15 Most Contained Node Arrangement on Sites

Combination	Support(%)
$C3 \cap E2 \cap F1 \cap G2$	100
$B1 \cap E2 \cap F1 \cap G2$	83
$A2 \cap E2 \cap F1 \cap G2$	67
$E2 \cap F1 \cap G2 \cap H2$	67

of attributes is small, the values obtained by GNP-KP and K-means are almost the same and higher than HC, however as the number of attributes increases, K-means shows lower values and HC shows higher values than K-means. GNP-KP shows the highest values in all the cases.

Log of GNP result explained by Table. 13, 14 and 15. Table. 13 shows nodes of each attribute which separated into three ranges and its coverage. Each attributes shows unbalanced coverage by each ranges. For example attribute F dominated with data with range between 11573 and 14458. This domination effects to attributes substances on each site(cluster) which shown by 14. Site with larger capacity contain more nodes variation of rule and data. There is no site that contain only one variation of rule. Its shown by always more than one variation of ranges for one attribute on each site. For example site 1 contain both of H1 and H2, so its possible to have a two variation of rule.

As described on section 3.1.3 about partial random, combination attribute range with high coverage also resulting high coverage. But because each site(cluster) have a limit of capacity, attribute range with high coverage become subtances of most sites or can be whole sites. Because high data amount difficult to be fit in one site's capacity, so remaining data will be distributed to other site. This case could be decreased by increase amount of attribute range, so its will decrease the range domination but its will increase calculation amount and impact to proccesing time.

Partial random created by combining nodes with high coverage which on average of coverage exceeds 500. Result of fixed combination shown by Table. 13. Support means ratio of combination to the whole six site. For example combination $C3 \cap E2 \cap F1 \cap G2$ available on every site so support are 100% and combination $B1 \cap E2 \cap F1 \cap G2$ available on five site so support is $5/6 = 83\%$. This feature could be used to explore patterns domination of records on dataset.

Table 13 Extracted Nodes and Its Coverage

	1		2		3	
A	[164,227]	160	[284,357]	583	[381,500]	257
B	[1129,1269]	522	[1456,1574]	173	[1651,2000]	305
C	[791,849]	418	[892,897]	10	[1020,1500]	572
D	[1,1]	135	[2,2]	499	[3,5]	366
E	[20,23]	148	[31,44]	631	[44,95]	221
F	[11573,14458]	687	[14569,15814]	184	[19105,30000]	129
G	[9898,10418]	304	[17664,21672]	637	[30634,50000]	59
H	[1,1]	244	[2,2]	264	[2,10]	492

Table 14 Nodes Substances On Each Site

Site	Attributes	Capacity	Data
1	{A2,B1,C3,D2,D3,E2,F1,G2,H1,H2}	100	69
2	{A1,A2,B1,B3,C2,C3,D2,D3,E1,E2,F1,G1,G2,H3}	500	209
3	{A3,C3,D2,D3,E2,F1,F2,G1,G2,H1,H2}	200	94
4	{A1,A2,B1,B3,C1,C3,D2,D3,E1,E2,E3,F1,F2,G2,G3,H1,H3}	600	420
5	{A3,B1,C3,D1,D3,E1,E2,F1,F3,G2,G3,H1,H2}	300	159
6	{A2,B1,B2,C1,C3,D1,D3,E1,E2,F1,F2,G1,G2,H2,H3}	200	49

5. CONCLUSIONS

This paper proposes novel clustering method combining Genetic Network Programming and Knapsack Problem to handle a record clustering. The proposed method can find good combinations of attributes to create rules for clustering, and also consider the capacity of sites to distribute rules. From the simulation results, the proposed method shows better clustering ability than K-means and hierarchical clustering in eight kinds of datasets. In the future, we will consider the replication problems of distributed databases, where the query frequency is considered in the clustering problems and frequently accessed data will be stored in several sites to decrease the load of the accesses.

REFERENCES

[1] Guinepain, S., Gruenwald, L., "Automatic Database Clustering Using Data Mining", *17th International Conference on Database and Expert Systems Applications (DEXA'06)*, 2006.

[2] P.R.Bhuyar, A.D.Gawande, A.B.Deshmukh, "Horizontal Fragmentation Technique in Distributed Database" *The International Journal of Scientific and Research Publications*, 2012.

[3] Sanjay Agrawal, V. Narasayya, B. Yang, "Integrating Vertical and Horizontal Partitioning into Automated Physical Database Design", *SIGMOD*, June 2004.

[4] 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*, Vol. 41, No. 1, pp. 130-139, 2011.

[5] Kaoru Shimada, Kotaro Hirasawa and Junglu Hu, Genetic Network Programming with Acquisition Mechanism of Association Rules, Journal of Advanced Computational Intelligence and Intelligence Informatics, Vol. 10, No. 1 pp. 102-111, 2006

[6] Jiangfei Zhao; TingLei Huang; Fei Pang; Yuanjie Liu, "Genetic Algorithm Based on Greedy Strategy in the 0-1 Knapsack Problem," *Genetic and Evolutionary Computing, 2009. WGEC '09. 3rd International Conference on , vol., no., pp.105,107*, 14-17 Oct. 2009

[7] Singh, R.P., "Solving 01 Knapsack problem using Genetic Algorithms", *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on , vol., no., pp.591,595*, 27-29 May 2011

[8] Elizondo-Leal, J.C.; Ramirez-Torres, G., "An Exact Euclidean Distance Transform for Universal Path Planning", *Electronics, Robotics and Automotive Mechanics Conference (CERMA), 2010 , vol., no., pp.62,67*, Sept. 28 2010-Oct. 1 2010

[9] Yanchi Liu, Zhongmou Li, Hui Xiong, Xuedong Gao, Junjie Wu, "Understanding of Internal Clustering Validation Measures Yanchi", *IEEE International Conference on Data Mining*, 2010.