Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2004 Proceedings

Americas Conference on Information Systems (AMCIS)

December 2004

Evaluating a Clique Partitioning Problem Model for Clustering High-Dimensional Data Mining

Haibo Wang
University of Mississippi

Bahram Alidaee
University of Mississippi

Gary Kochenberger
University of Colorado at Denver

Evaluating a Clique Partitioning Problem Model for Clustering High-Dimensional Data Mining

Haibo Wang
University of Mississippi
chwang@olemiss.edu

Bahram Alidaee
University of Mississippi
balidaee@bus.olemiss.edu

Gary Kochenberger
University of Colorado at Denver
Gary.kochenberger@cudenver.edu

ABSTRACT

This paper considers the problem of clustering high dimensional data as a clique partitioning problem. Data objects within a cluster have high degree of similarity. The similarity index values are first constructed into a graph as a clique partitioning problem which can be formulated into a form of unconstrained quadratic program model and then solved by a tabu search heuristic incorporating strategic oscillation with a critical event memory. Results from other clustering techniques are compared on a set of instances from open literatures. The computational results highlight the robustness of this new model and solution methodology.

Keywords

Clustering; High-Dimensional; Nominal-Scaled; Data Mining; Tabu Search; Strategic Oscillation; Clique Partitioning Problem; Unconstrained Binary Quadratic Programming.

INTRODUCTION

Data mining is the scientific method to extract useful information from large data sets or databases and recognized as a powerful tool for knowledge discovery. Clustering in data mining analyzes data objects without consulting a known class label. Objects within a cluster are similar to the characterization because in the cluster object all have high degree of similarity.

Many business decisions depend on the results from a large amount of data which are often high-dimensional and some are nominal-scaled. The high-dimensional nominal-scaled data record poses unique challenge to the traditional clustering techniques. For example, customer relationship management applications are very important to many business decisions and they are considered as essential enterprise business strategy. Clustering customers by their buying behavior, which is often high-dimensional data record, can produce useful knowledge for marketing decision support and finding new opportunities with existing customers. The clustering result can provide competitive advantage and improve efficiencies and effectiveness in marketing, sales, transportation policy and customer support. It can help discover customer shopping patterns and trends. An efficient clustering method for the high-dimensional customer buying behavior data record can provide decision makers timely information to improve the quality of customer service.

In general, it is difficult to cluster high-dimensional nominal-scaled data. The pattern or function estimate found in a one-dimensional space is often unable to scale up well in high dimensional space. The amount of data needed to represent the pattern with the same level of accuracy often increases exponentially. This difficult is also known as "curse-of-dimensionality". Many techniques used to cluster low-dimensional data become ineffective in high-dimensional space. The measure used in the low-dimensional data such as Euclidean distance can not be used to high-dimensional nominal-scaled data. Besides, there are often many clusters in the high-dimensional data. From the computational complexity point of view, it is an NP-hard problem (Brucker, 1978; Welch, 1982) if there are more than 3 clusters in the data.

There are two types of models in clustering data: hierarchical and partitional models. The difference between two models is the number of clusters in the results. Hierarchical model can also be represented as a multi-level tree structure or minimum spanning tree structure. It produces the cluster result by choosing certain threshold value in tree structure. The applications for hierarchical model are generally loosely coupled with small or medium size. The model based on the hierarchical clustering have an irreversible limitation, which means when two objects are grouped together at some points, there is no way to retrace the steps even if it leads to suboptimal clustering results at the end. This disadvantage precludes formation of better clusters at later stages. Hierarchical clustering methods often produce cluster quickly with a lower degree of separation. The applications for partitioning model are generally large and computationally difficult to construct the multi-level tree structure due to the tight-coupled relationship among the objects.

The techniques for hierarchical model are based on the linkage between the groups of the data objects. The techniques for partitioning model are mostly based on the formulation of local or global objective function. Combinatorial optimization approaches are often applied to obtain the optimum solution of the objective function. The result clusters are the configurations of the best solution with multiple runs. There are many heuristic approaches to find the optimum solution. Some researchers studied the performance of a group of heuristic methods for clustering and found out tabu search having the best performance on a small continuous data set (Al-Sultan and Khan, 1996; Mishra and Raghavan, 1994).

From a methodology point of view, clustering encompasses the methods and techniques from databases, machine learning, statistics, artificial intelligence and optimization. In this study, clustering data mining problem is reformulated as clique partitioning problem (CPP) by constructing a graph from similarity value and a threshold value. Since CPP is NP-hard in general, heuristic methods are needed to solve large sized problems. The associated CPP model is solved by first re-casting it into the form of unconstrained binary quadratic programming (UBQP), which is then solved by a tabu search heuristic.

CLIQUE PARTITIONING PROBLEM MODEL AND SOLUTION METHODOLOGY

The clique partitioning problem is also known as the uncapacitated clustering problem and it can be described as: Given a complete graph $G = G(V, E)$ on n objects with each object as a node and the similarity score between a pair of objects as edge weight w_{ij}, a cluster in G is the subset of nodes together as complete sub-graph. The weight of a cluster is the sum of the weights of the edge (similarity scores) within the cluster. The cluster problem is to partition the nodes into clusters so that the sum of the weights of the intra-clusters is maximized. In the mean time, the total weight of inter-clusters is minimized.

The standard mathematical model for CPP (Grötschel and Wakabayashi, 1989) is formulated as following:

$$\max \quad \sum_{(i,j)\in E} W_{ij} X_{ij}$$

$$st$$

$$x_{ij} + x_{ir} - x_{jr} \le 1 \quad \forall \; all\ distinct\ i, j.r \in V$$

$$x_{ij} \in \{0,1\}$$

The constraints are also known as triangle inequalities (Grötschel and Wakabayashi, 1990). The number of variables in this model is the number of edges: $\frac{n(n-1)}{2}$ and the number of variables is considerable large in this linear program model. The number of constraints for the standard formulation is: $3\,C_n^3$ or $\frac{n(n-1)(n-2)}{2}$. Due to a large number of the constraints with this model, there are many algorithms applied to lift the facet-defining inequalities. Grötschel and Wakabayashi (1989) introduced a cutting plane algorithm. They only used a subset of constraints initially and only add new constraints if they are

violated by the linear programming solution. In fact, a small portion of constraints are actually used to find the optimal solution for small-sized problem. However, such model is undesirable to large-sized problems because it is very difficult to add constraints into the problem without introducing more new variables.

The new model proposed in this study is formulated as:

$$\max \quad \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_{ij} \sum_{k=1}^{k\_\max} x_{ik}\, x_{jk}$$

$$st$$

$$\sum_{k=1}^{k\_\max} x_{ik} = 1 \qquad i = 1, \ldots, n$$

$$x_{ik} \in \{0,1\}$$

where $k\_\max$ is the maximum number of cliques allowed in the solution, w_{ij} is edge weight and x_{ik} is equal to 1 if node i is assigned to clique k and 0 otherwise. The number of variables in this model is $n \times k\_\max$. The number of constraints is $\boldsymbol{n}$ in this constrained quadratic program (CQP) model, which is much smaller than the standard linear program model. Some researchers have developed a mechanism to reduce the number of constraints in CQP model by introducing a penalty functions and convert CQP model into an UBQP model (Glover, Kochenberger, Alidaee and Amini, 1999; Beasley, 1998). In general, the standard constrained quadratic problem can be formulated as:

$$CQP \quad \max f(x) = xQx$$
$$st \quad Ax = b$$
$$x_i \in \{0,1\}$$

This standard CQP formulation can represent both quadratic and linear problems and Q is a diagonal matrix for simple linear problems. Since $x_j^2 = x_j$ for any binary variables, the linear term in the objective function can be transformed into a quadratic term. This constrained quadratic program model can be converted into unconstrained quadratic binary program model by introducing an associated quadratic infeasibility penalty functions $\mathrm{P}(Ax-b)^t(Ax-b)$ to handle the constraints. Then the CQP model can be transformed as:

$$\begin{aligned} UBQP \quad & \max f(x) = xQx - \mathrm{P}(Ax-b)^t(Ax-b) \\ & = xQx - xDx + Cx + S \\ & = x\hat{Q}x \\ & x_i \in \{0,1\} \end{aligned}$$

where

$$D = \mathrm{P}A^tA,$$
$$C = 2\mathrm{P}b^tA$$
$$S = -\mathrm{P}b^tb$$

Since constant S has no contribution to the optimal solution, it is easy to get $\hat{Q} = Q - D + C$. The proposed CQP model in this study has equality constraints. It is easy to formulate the new model into UBQP $x\hat{Q}x$ by inserting penalized equality constraints into objective function with right-hand side value as 1. However, it is difficult to formulate the standard linear program model for CPP into UBQP $x\hat{Q}x$ model with inequality constraints, which means the large number of bounded slack variables needed to be inserted into each constraint.

The following equation is an example for 5 nodes and 2 cliques with the proposed model:

$$\begin{aligned}\textbf{max } f(x) = {} & w_{12}(x_{11}x_{21} + x_{12}x_{22}) + w_{13}(x_{11}x_{31} + x_{12}x_{32}) + w_{14}(x_{11}x_{41} + x_{12}x_{42}) \\ & + w_{15}(x_{11}x_{51} + x_{12}x_{52}) + w_{23}(x_{21}x_{31} + x_{22}x_{32}) + w_{24}(x_{21}x_{41} + x_{22}x_{42}) \\ & + w_{25}(x_{21}x_{51} + x_{22}x_{52}) + w_{34}(x_{31}x_{41} + x_{32}x_{42}) + w_{35}(x_{31}x_{51} + x_{32}x_{52}) \\ & + w_{45}(x_{41}x_{51} + x_{42}x_{52})\end{aligned}$$

st

$$\begin{aligned} x_{11} + x_{12} &= 1 \\ x_{21} + x_{22} &= 1 \\ x_{31} + x_{32} &= 1 \\ x_{41} + x_{42} &= 1 \\ x_{51} + x_{52} &= 1 \end{aligned}$$

$x_{ij} \in \{0,1\}$ ***for*** $i = 1,2,3,4,5$ ***and*** $j = 1,2$

The UBQP transformation for this example is:

$$\textbf{max } f(x) = x\hat{Q}x^t$$

where

$$x = [x_{11}\ x_{12}\ x_{21}\ x_{22}\ x_{31}\ x_{32}\ x_{41}\ x_{42}\ x_{51}\ x_{52}] \quad x^t = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{21} \\ x_{22} \\ x_{31} \\ x_{32} \\ x_{41} \\ x_{42} \\ x_{51} \\ x_{52} \end{bmatrix}$$

$$\hat{Q} = \begin{bmatrix}
P & -P & w_{12} & 0 & w_{13} & 0 & w_{14} & 0 & w_{15} & 0 \\
-P & P & 0 & w_{12} & 0 & w_{13} & 0 & w_{14} & 0 & w_{15} \\
w_{12} & 0 & P & -P & w_{23} & 0 & w_{24} & 0 & w_{25} & 0 \\
0 & w_{12} & -P & P & 0 & w_{23} & 0 & w_{24} & 0 & w_{25} \\
w_{13} & 0 & w_{23} & 0 & P & -P & w_{34} & 0 & w_{35} & 0 \\
0 & w_{13} & 0 & w_{23} & -P & P & 0 & w_{34} & 0 & w_{35} \\
w_{14} & 0 & w_{24} & 0 & w_{34} & 0 & P & -P & w_{45} & 0 \\
0 & w_{14} & 0 & w_{24} & 0 & w_{34} & -P & P & 0 & w_{45} \\
w_{15} & 0 & w_{25} & 0 & w_{35} & 0 & w_{45} & 0 & P & -P \\
0 & w_{15} & 0 & w_{25} & 0 & w_{35} & 0 & w_{45} & -P & P
\end{bmatrix}$$

This simple example illustrates the potential applicability of the transformation for many quadratic and linear program problems.

Tabu Search

A tabu search heuristic method is used to solve the unconstrained quadratic program model in this study and it exploits the tabu search memory on the unconstrained quadratic program problem by incorporating strategic oscillation with a critical event memory. Tabu search directs and manages the search process by incorporating short-term and long-term memories to explore the forbidden regions in the search space. The two key features in tabu search are intensification and diversification technique. In general, the intensification approach is to modify the choice rule for a new combination of moves and can be used to perform a more thorough search in the favorable region while diversification approach is used to explore the new otherwise forbidden regions and may generate a new solution which cannot be found in the intensification approach. In intensification approach, the tabu list is used to keep a list of good solutions. The short term memory is associated with intensification approach and the long term memory is a combination of intensification and diversification and it is often referred as adaptive memory. Such implementation keeps track of the result on most recent moves and records the path of finding recent result. This feature can not be found in other local search methods such as simulated annealing. The performance of tab search is associated with the size of tabu list where the larger size of tabu list has a better chance to find good solutions but small tabu list may produce very good results in some applications. The size of tabu list depends on the problem itself. In this study, the size of tabu list is set to 7.

Strategic Oscillation

Strategic Oscillation is the diversification technique in tabu search (Lakkentangen and Glover; 1998; Dowsland, 1998). It can start from a feasible solution region and cross back and forth between feasible and infeasible regions. The method switches the search process direction between two phases: constructive and destructive. In constructive phase, the search process is accomplished by switching the value of setting variables from 0 to 1. In destructive phase, it switches the value of setting variables from 1 to 0 gradually.

The basic step of strategic oscillation is:

1. Start from a feasible solution region. Perform moves to get new solutions until hitting a boundary of infeasible solution region.

2. Change the evaluation criterion or extend neighborhood function to allow search process crossing such boundary.

3. Go further beyond the boundary for certain depth which can be static or random.

4. Then turn around to enforce feasibility and cross back.

5. Repeat steps above and crossing boundary from different directions until the no better solution can be found or reach the limit of iterations.

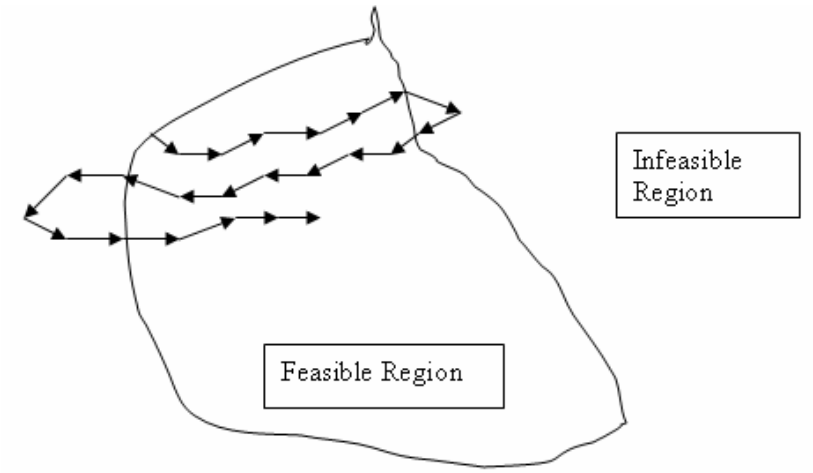

Figure 1. Illustration of Simple Strategic Oscillation

Critical Event Memory

The critical event memory is constructed when the search process reach a stage, which is referred as critical event where it generates a subclass of locally optimal solution. It controls the search process by adding penalty to the favorable moves in the constructive phase and keeps the attractive moves in the destructive phase. It uses a parameter span to incorporate the critical event memory into strategic oscillation. The value of span is set to 1 at the beginning of the search process and is increased to a preset limiting value. When the value of span reaches its limit, it is decreased gradually until it reaches the initial value of 1 where it is counted as a complete cycle.

RE-CAST CLUSTERING DATA MINING PROBLEM INTO CPP

There are four steps in the approach of clustering data on the high-dimensional space in this study:

1. Transform the high dimensional nominal-scaled data record into binary representation and calculate the similarity index and similarity scores of the objects.
2. Re-cast the problem into CPP by constructing a graph from similarity scores.
3. Solve the CPP in a form of an UBQP model with a tabu search heuristic.
4. Analyze output and abstract knowledge from the data.

This approach provides some advantages over other clustering techniques since the high-dimensionality of data does not affect other steps when the similarity scores are calculated at step 1. The CPP model provides the guarantee of equally importance among the resulting clusters.

Many high-dimensional data are nominal-scaled and very difficult to be clustered. Extensive techniques have been developed to reduce the dimensions and solved the problem by linear programming models or by integer programming models on a lower dimensional space. These approaches are also known as feature selection based on the assumption that the attributes are class conditionally or cluster conditionally dependent.

The nominal-scaled high dimensional data contains binary and non-binary data record. If the number of distinct values of the attribute is two, the data can be treated as either binary data by converting one value in the attribute as ‘1’ and the other as ‘0’ or regular nominal-scaled data by splitting one original attribute into two binary attributes. Some data records have a mix of

binary attributes and non-binary attributes. Hamming Distance is the standard measure for binary data record (Illingworth, Glaser and Pyle, 1983)

There are two approaches to the binary representations in this study by either comparing a pair of objects with logical bit or converting the level of attribute values into multiple binary attributes. The size of binary representation of the first approach is generally larger than the second approach. In the logical bit approach, when two objects have the same values on the same attribute, the logical bit is 1 and 0 otherwise.

The following example shows how to transform a data record with original non-binary attribute into its binary representation.

ID	Color	Size
1	YELLOW	LARGE
2	YELLOW	SMALL
3	RED	SMALL
4	YELLOW	LARGE
5	RED	SMALL
6	BLUE	LARGE

If user chooses logical bits, the binary representation for this data set will be:

Objects	Color	Size
1,2	1	0
1,3	0	0
1,4	1	1
1,5	0	0
1,6	0	1
2,3	0	1
2,4	1	0
2,5	0	1
2,6	0	0
3,4	0	0
3,5	1	1
3,6	0	0
4,5	0	0
4,6	0	1
5,6	0	0

If the user uses the multiple attributes approach, the level of attribute values need to computed at first in order to decide the number of binary attributes. There are three distinct values in the Color attribute and the script transforms this attribute to three new binary attributes. There are two distinct values in the Size attribute. The script transforms it to two new binary attributes for the Size attribute or transforms it into one binary attribute. A binary representation for the data set is:

ID	C-YELLOW	C-RED	C-BLUE	Size

1	1	0	0	1
2	1	0	0	0
3	0	1	0	0
4	1	0	0	1
5	0	1	0	0
6	0	0	1	1

For the data-record with all non-binary attribute, some researchers (Erlich, Gelbard and Spiegler, 2003) proposed a Positive Attribute Distance (PAD) for similarity measurement. They (Erlich, Gelbard and Spiegler, 2002) also proposed a Pair Similarity Index (PSI) for similarity measurement.

This study formulates a similarity index with the original attribute distance (OAD). Like PAD and PSI, it uses a binary representation of the existence or absence of an attribute in a given object being observed. The resulting binary string representing the object is then used to calculate distance to other strings using the value 1 for '1' bit and 0 for '0' bit on the non-binary attributes, exclusive OR logical operation and the total number of original attributes. For the binary representation of each object, there is a binary string $s = s_{ba} + s_{nba}$, where s_{ba} is the substring of $\boldsymbol{s}$ for original binary attributes, and s_{nba} is the substring of s for original non-binary attributes. The OAD similarity index (S_n) is defined as:

$$S_n = \frac{V_{ba} + V_{nba}}{N_{na}}$$

$$V_{ba} = L_{ba} - HD(s_{ba1}, s_{ba2})$$

$$HD(s_{ba1}, s_{ba2}) = s_{ba1} \oplus s_{ba2}$$

where V_{nba} is the sum of the values of '1' and '0' bit on both s_{nba1} and s_{nba2} when they appear on the same location. For the logical bit approach, V_{nba} is the sum of the values of '1' and '0' bit for the pair of objects where V_{ba} is equal to zero for logical bit binary representation. V_{ba} is the Hamming Distance similarity index values for substring s_{ba}. L_{ba} is the length of substring s_{ba}. $HD(s_{ba1}, s_{ba2})$ is the Hamming Distance between substrings s_{ba1} and s_{ba2}, $\oplus$ denotes the logical operation XOR. N_{na} is the total number of the original nominal-scaled attributes. Table 1 shows the similarity index values from different measures.

The term similarity score bases on the notation of similarity indexes. If the value of SI_{OAD} is 1, two objects are identical. In another word, these two objects have 100% degree of similarity. If the value of SI_{OAD}, two objects have absolute diversity. These two objects have 0% degree of similarity. Similarity Threshold (ST) can be chosen by the user based on their domain knowledge and computational experiences. Computational result shows a good choice of ST with the average value of SI_{OAD} for all objects in the data set. If the similarity index of a pair of objects is larger than the value of ST, these two objects have a high degree of similarity and are intended to be grouped into a cluster. Otherwise they have less likelihood to be grouped into a cluster. The Similarity Score (SS) is the percentage value of the difference between a degree of similarity of a pair of objects and the similarity threshold. After the similarity indexes are calculated from nominal-scaled data record with SI_{OAD}, it is transformed into similarity score.

Objects Pair	**Logical Bit**	**Multiple-Attributes**			
	SI_{OAD}	SI_{HD}	SI_{PAD}	SI_{PSI}	SI_{OAD}
1,2	0.67	0.8	0.8	0.67	0.67
1,3	0	0.2	0	0	0
1,4	1	1	1	1	1
1,5	0.33	0.4	0.4	0.33	0.33
1,6	0.33	0.4	0.4	0.33	0.33
2,3	0.33	0.4	0	0	0.33
2,4	0.67	0.8	0.8	0.67	0.67
2,5	0.67	0.6	0.5	0.5	0.67
2,6	0	0.2	0	0	0
3,4	0	0.2	0	0	0
3,5	0.67	0.8	0.67	0.5	0.67
3,6	0.33	0.4	0	0	0.33
4,5	0.33	0.4	0.4	0.33	0.33
4,6	0.33	0.4	0.4	0.33	0.33
5,6	0	0.2	0	0	0

Table 1. Similarity Index Values for the Two Binary Representations with Different Index Models

$$ST = \overline{SI}_{OAD} = \frac{\sum_{i=1}^{k} SI_{OAD}(i)}{k}$$

$$SS_{s1,s2} = (SI_{OAD_{s1,s2}} - ST) * 100$$

$$k = \frac{n(n-1)}{2}$$

With the similarity threshold approach, the low level of similarity will have a negative similarity score value, which will separate the objects with low level of similarity into different groups (clusters). If many objects have low level of similarity, the threshold value will be small and some objects will have positive similarity score value even though their index values are low. This approach can identify the objects with low similarity value easily. These objects are known as outliers, which is very important to some application such as DNA-array study. The new approach will identify the outliers and still keep other objects in each cluster.

The graph input for the previous data set with SS values can be constructed as:

Node1	Node2	Weight
1	2	29
1	3	-38
1	4	62
….		
5	6	-38

CLUSTERING DATA MINING RESULTS

To evaluate the solution quality of CPP model, this study chooses a set of data from recent studies (Erlich, Gelbard and Spiegler, 2002; Erlich, Gelbard and Spiegler, 2003) with comparable results from the literatures. This study adopts the measure of within-cluster similarity and between-cluster similarity defined by Erlich, Gelbard and Spiegler (2002). This study uses the average similarity index value as the threshold based on the computational experiences. The similarity index values for the data set from the literature are included at the supplement.

k_max	ST Value	Number of Clusters	Average Within-Cluster Similarity	Average Between-Cluster Similarity
15	0.2	3	0.309	0.192
15	0.22	4	0.397	0.143
15	0.3	7	0.473	0.091
15	0.4	10	0.479	0.077
15	0.5	13	0.578	0.064
4	0.2	3	0.309	0.192
4	0.22	4	0.397	0.143
4	0.3	4	0.346	0.159
4	0.4	4	0.311	0.161
4	0.5	4	0.288	0.164
3	0.3	3	0.232	0.175
4	0.3	4	0.346	0.159
5	0.3	5	0.358	0.138
6	0.3	6	0.417	0.116
7	0.3	7	0.473	0.091
3	0.4	3	0.242	0.177
4	0.4	4	0.311	0.161
5	0.4	5	0.336	0.142
6	0.4	6	0.349	0.139
7	0.4	7	0.397	0.127

Table 2. Correlation Among Values of k_max, ST Value, Number of Clusters, Average Within-Cluster Similarity and Average Between-Cluster Similarity

Table 2 shows the number of clusters depending on the value of K_max and ST. When K_max is large enough and fixed, the number of clusters is decided by the value of ST and there is a positive correlation between them. In the meantime, the average within-cluster similarity increases and the average between-cluster similarity decreases when the value of ST increases. The smaller clusters have better compactness and more separation between each other.

When the K_max value is small and fixed, the positive correlation disappears. The number of clusters is decided by the value of K_max instead of the value of ST. The average within-cluster similarity value decrease and the average between-cluster similarity increases if the value of ST is larger than the average value of similarity index value, which is equal to 0.22 in the data set. The choice of ST will provide the capability of splitting or merging of clusters and it helps to guide the researchers toward the right number of clusters.

When the K_max value is various and the value of ST is larger than the average value of similarity index value, the number of cluster is decided by K_max and there is a positive correlation between K_max and the average within-cluster similarity value.

Methods	**Cluster**	**Size**	**Cluster Output**
1	1	24	1,6,7,10,11,15,17,18,19,20,21,22,23,24,25,26,27,28,33,34,35,36,37,38
	2	8	2,3,4,9,29,32,39,40
	3	5	5,12,13,14,16
	4	3	8,30,31
2	1	12	1,4,8,9,17,23,25,29,30,31,32,37
	2	11	5,6,10,11,13,14,15,16,22,27,36
	3	9	2,3,18,19,20,35,38,39,40
	4	8	7,12,21,24,26,28,33,34
3	1	21	5,10,11,12,13,14,15,16,18,19,20,21,22,24,26,27,28,33,35,36,38
	2	6	1,8,17,23,25,37
	3	8	4,9,29,30,31,32,39,40
	4	5	2,3,6,7,34
4	1	24	1,6,7,10,11,12,15,17,18,19,20,21,22,23,24,25,26,27,28,33,34,35,36,38
	2	5	2,3,4,9,40
	3	4	5,13,14,16
	4	7	8,29,30,31,32,37,39
5	1	15	10,15,16,18,19,20,21,24,27,28,33,34,35,36,38
	2	6	1,2,17,22,23,25
	3	9	3,5,6,7,11,12,13,14,26
	4	10	4,8,9,29,30,31,32,37,39,40
6	1	21	5,10,11,12,13,14,15,16,18,19,20,21,24,26,27,28,33,34,35,36,38
	2	5	1,17,22,23,25
	3	4	2,3,6,7
	4	10	4,8,9,29,30,31,32,37,39,40

7	1	19	1,6,7,10,15,17,19,20,21,22,23,24,25,27,28,33,35,36,38
	2	6	2,3,4,9,34,40
	3	7	5,11,12,13,14,16,26
	4	8	8,18,29,30,31,32,37,39
8	1	13	10,11,12,13,14,15,16,22,24,26,27,28,36
	2	12	1,8,9,17,23,29,30,31,32,37,39,40
	3	10	2,3,4,5,18,19,20,25,35,38
	4	5	6,7,21,33,34
9	1	15	10,15,18,19,20,21,22,24,25,27,28,33,35,36,38
	2	12	2,3,5,6,7,11,12,13,14,16,26,34
	3	6	4,9,29,32,39,40
	4	7	1,8,17,23,30,31,37

Table 3. Cluster Results by Different Techniques and Software

The techniques listed in Table 3 are: 1. CPP model with tabu search, 2. clustering analysis tool from SAS, 3. positive attribute distance with a local search algorithm, 4. furthest neighbor method, 5. Ward's method with SPSS, 6. within groups method, 7. centroid method with SPSS, 8. centroid method with SYSTAT, 9. Ward's method with SYSTAT. See Erlich, Gelbard and Spiegler (2003) for the detail descriptions of the methods 2-9 and the cluster output. Each method generates 4 clusters for the data set. The average within- cluster similarity value and the average between-cluster similarity value for methods 2-9 are also reported. Erlich, Gelbard and Spiegler (2002) reported an improvement on within-cluster similarity value by method 3 and did not compare the between-cluster similarity among the methods. Table 4 shows the average between-cluster similarity value and average within-cluster similarity value by all methods.

Methods	**Average Between-Cluster Similarity Value**	**Average Within-Cluster Similarity Value**
1	0.143	0.397
2	0.210	0.225
3	0.180	0.334
4	0.150	0.323
5	0.180	0.258
6	0.180	0.332
7	0.158	0.312
8	0.209	0.287
9	0.170	0.318

Table 4. Cluster Qualities by Different Techniques and Software

ANALYZE OUTPUT AND VALIDATE CLUSTERS

The quality of clustering depends on two aspects: with-cluster compactness and between cluster separation. A better clustering result means an increase in similarity among the objects in the same cluster and a decrease in similarity among the objects between any two different clusters. Some researchers do not address cluster separation, which is inadequate and inappropriate as performance. The previous study reported the average similarity value within the cluster and between the clusters. To prove their results better than other methods, they did comparison on within cluster similarity. They did not compare the similarity between clusters since their results did not show any advantages in fact it is worse than some methods they compared. They used a hierarchical clustering approach which often produce cluster with weak separation. When we analysis the results, we found the ratio between within-cluster similarity and between-cluster similarity is better and more convincing than the difference between these two. For example, one method produce 0.4 for within-cluster similarity and 0.2 for between-cluster similarity, another method produce 0.35 for within-cluster similarity and 0.15 for between-cluster similarity. If we use difference between two similarities as the measure for quality, both methods produce the same quality. If we use ratio value, the second method produces a better quality. In fact, it is hard to get a better separation in clustering data. This study uses the ratio of within-cluster similarity vs between-cluster similarity as cluster efficiency to measure the quality of clustering techniques. Table 5 shows the model and methodology in this study produce better clustering results comparing to other clustering techniques reported in the literatures (Erlich, Gelbard and Spiegler, 2003) Erlich, Gelbard and Spiegler, 2003). The new approach produces cluster with strong structure.

Methods	Cluster Efficiency
1	2.776
2	1.071
3	1.856
4	2.153
5	1.433
6	1.844
7	1.975
8	1.373
9	1.871

Table 5. Cluster Efficiency on the Groups Produced by Various Clustering Techniques

CONCLUDING REMARKS

Clustering analysis is still the most popular tool in data mining (reported by KDNuggets in 2001). This study addresses the task of identifying the clusters on the basis of high dimensional data set. It seeks to overcome the limitations of the recent studies through the development of a methodology based on a partitional clustering approach. The new approach in this study is implemented by the introduction of new model of CPP, an appropriate similarity measure and the development of a meaningful measure for evaluating and comparing the solutions. This study uses a simple and robust similarity index value formulation to transform the data from high-dimensional space into the similarity space, which capture the degree of similarity of any given pair of objects. The value of similarity threshold allows one to determine the appropriate similarity measure based on the domain knowledge or a given situation. Similarity threshold is a common approach of hierarchical clustering methods. The new approach in this study combines the methodologies from both hierarchical and partitional clusters methods. The hierarchical clustering methods often produce clustering results quickly with a lower quality and the partitional clustering methods often produce high quality clusters with the relative difficulty to be implemented. The new approach presents a solution to cluster data with both speed and quality with easier implementation.

There is no standard to validate cluster and evaluate output, it varies from application to application. A desirable interpretation of the output requires a balance of the implicit assumption and interesting information abstracted from the clusters. The cluster efficiency reflects two important aspects in clustering: the compactness within the cluster and the separation among clusters.

The model and methodology in this paper can easily handle the data with thousands of objects with thousands of attributes, which means millions of patterns. The computational time depends on the number of objects and k_max only, does not depend on the number of attributes. Such approach has increased power when the number of attribute is continuously increasing.

ACKNOWLEDGMENTS

The authors express their gratitude to the Mississippi Center for Supercomputing Research for providing computing resource.

REFERENCES

1. Al-Sultan, K. S. and Khan, M. M. (1996) Computational experience on four algorithms for the hard clustering problem, *Pattern Recognition Letters*, 17, 295-308.
2. Beasley, E. (1998) Heuristic Algorithms for the Unconstrained Binary Quadratic Programming Problem, Working Paper, Imperial College.
3. Brucker, P. (1978) On the complexity of clustering problems, In M. Beckman and H.P. Kunzi, editor, *Optimization and Operations Research: Lecture Notes in Economics and Mathematical Systems*, 45--54. Springer-Berlin.
4. Dowsland, K. (1998) Nurse scheduling with tabu search and strategic oscillation. *European Journal of Operational Research*, 106, 393-407.
5. Erlich, Z., Gelbard, R. and Spiegler, I. (2002) Data mining by means of binary representation: a model for similarity and clustering, *Information Systems Frontiers*, 4, 2, 187-197.
6. Erlich, Z., Gelbard, R. and Spiegler, I. (2003) Evaluating a positive attribute clustering model for data mining, *Journal of Computer Information Systems*, 43, 3, 100-108.
7. Glover, F. and Laguna, M. (1997) Tabu Search, Kluwer Academic Publishers.
8. Glover, F., Kochenberger, G. and Alidaee, B. (1998) Adaptive memory tabu search for binary quadratic programs, *Management Science*, 44, 3, 336-345.
9. Glover, F., Kochenberger, G., Alidaee, B. and Amini, M. (1999) Tabu search with critical event memory: an enhanced application for binary quadratic programs In: *Meta-Heuristics, Advances and Trends in Local Search Paradigms for Optimization*, S. Voss, S. Martello, I. Osman, and C. Roucairol (Eds.), Kluwer Academic Publishers.
10. Grötschel, M. and Wakabayashi, Y. (1989) A cutting plane algorithm for a clustering problem, *Mathematical Programming*, 45, 59-96.
11. Grötschel, M. and Wakabayashi, Y. (1990) Facets of the clique partitioning polytope, *Mathematical Programming*, 47, 367-387.
12. Illingworth, V., Glaser, E. L. and Pyle I. C. (1983) Hamming Distance, *Dictionary of Computing*, Oxford: Oxford University Press, 162-163
13. Lakkentangen, A. and Glover F. (1998) Solving Zero-one Mixed Integer Programming Problems Using Tabu Search. *Eurpean Journal of Operational Research*, 106, 624-658.
14. Mishra, S. K. and Raghavan, V. V. (1994) An empirical study of the performance of heuristic methods for clustering, *Pattern Recognition in Practice*, E. S. Gelsema and L.N. Kanal Eds., North Holland, 425-436.
15. Welch W. J. (1982) Algorithmic complexity three NP-hard problems in computational statistics, *Jour. Stat. Comp. and Sim.* 15, 17-25

SUPPLEMENT

Erlich, Z., Gelbard, R. and Spiegler, I. (2002) Data mining by means of binary representation: a model for similarity and clustering, *Information Systems Frontiers*, 4, 2, 187-197

node	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1		0.13	0.13	0	0.13	0.25	0.25	0.25	0.25	0.13	0.13	0	0.25	0.13	0.25	0	0.5	0.25	0.25	0.13	0.25	0.13	0.38	0.13	0.25	0.13	0	0.25	0	0.25	0.13	0	0.25	0.25	0.25	0.25	0.38	0.25	0	0
2			0.5	0.38	0.25	0.38	0.5	0.25	0.25	0.13	0.13	0.13	0.25	0	0.13	0	0.13	0.25	0.13	0.13	0.38	0.13	0.25	0	0.13	0.13	0	0.13	0.38	0	0.13	0.13	0.13	0.38	0.25	0.13	0.25	0.25	0.5	0.38
3				0.13	0	0.13	0.25	0.13	0.25	0.13	0.13	0.13	0.13	0.25	0.13	0	0	0.25	0	0.38	0.38	0.13	0.13	0.13	0	0.13	0.13	0	0.25	0	0.13	0.25	0.13	0.25	0.13	0.13	0.13	0.25	0.38	0.13
4					0.25	0	0	0.25	0.5	0	0.13	0.38	0.25	0	0	0	0.13	0	0.13	0	0	0	0.13	0	0.13	0	0.13	0	0.5	0	0	0.25	0	0.13	0.25	0	0.13	0	0.25	0.38	
5						0.25	0.25	0.25	0	0.25	0.25	0.38	0.38	0.25	0.13	0.25	0	0.13	0.25	0	0	0.13	0	0.25	0	0.13	0.13	0.13	0.13	0.25	0.13	0.13	0.13	0	0.25	0.25	0.13	0.38	0	0.13		
6							0.63	0	0	0.5	0.13	0.13	0.25	0.13	0.38	0.13	0.25	0.25	0.38	0	0.25	0.25	0.25	0.13	0.38	0.25	0.25	0.25	0.13	0	0.25	0.13	0.5	0.13	0.25	0.25	0.25	0.38	0.25	0.13		
7								0	0.13	0.25	0.25	0.25	0.25	0.25	0.25	0.13	0.25	0.25	0.25	0.13	0.25	0.13	0.38	0.25	0.25	0.38	0.13	0.13	0.13	0	0.13	0	0.38	0.38	0.25	0.38	0.13	0.38	0.13	0
8									0.25	0	0.38	0.13	0	0	0.13	0	0.13	0.13	0.13	0.13	0.13	0	0.25	0.13	0.13	0.13	0	0.13	0.25	0.5	0.38	0.25	0.13	0.25	0.13	0.13	0.5	0.13	0.25	0.25
9										0	0.13	0	0.25	0.38	0.13	0	0.38	0	0	0.13	0.13	0	0.25	0.13	0	0	0	0.13	0.38	0.13	0	0.25	0.25	0.25	0.25	0.13	0.25	0	0.25	0.25
10											0.25	0.25	0.13	0.25	0.63	0.25	0.25	0.38	0.63	0.25	0.38	0.5	0.25	0.38	0.38	0.38	0.5	0.38	0.13	0.13	0.25	0.38	0.63	0.13	0.5	0.5	0.25	0.63	0.25	0.13
11												0.5	0.25	0.25	0.38	0.25	0.13	0.25	0.13	0.25	0.25	0.13	0.13	0.5	0	0.5	0.38	0.13	0.13	0.13	0	0	0.38	0.25	0.38	0.5	0.13	0.25	0	0.13
12													0.38	0.38	0.38	0.13	0	0.13	0.13	0.25	0.25	0.13	0	0.5	0	0.38	0.25	0.13	0.13	0	0	0	0.25	0.13	0.38	0.5	0	0.25	0	0.25
13														0.38	0.13	0.13	0.13	0	0.13	0	0	0	0	0.13	0	0.25	0.25	0	0.25	0	0	0	0.13	0	0.25	0.13	0	0.13	0.13	0.25
14															0.25	0.25	0.13	0.13	0.13	0.13	0.25	0.13	0.25	0	0.25	0.38	0	0	0	0	0	0.38	0	0.38	0.25	0	0.25	0	0.13	
15																0.25	0.5	0.38	0.5	0.5	0.5	0.63	0.38	0.63	0.5	0.38	0.5	0.5	0.13	0.25	0.25	0.25	0.63	0.25	0.63	0.75	0.38	0.5	0.25	0.13
16																	0.13	0.25	0.38	0.13	0	0.25	0.13	0.25	0.13	0.13	0.25	0.13	0.25	0.13	0.25	0.13	0.13	0.13	0.38	0.25	0	0.25	0.25	0.13
17																		0.25	0.25	0.25	0.25	0.5	0.63	0.25	0.63	0.13	0.13	0.38	0.13	0.25	0.13	0.13	0.25	0.25	0.38	0.38	0.38	0.25	0.13	0
18																			0.38	0.38	0.25	0.38	0.25	0.25	0.25	0.25	0.25	0.38	0.38	0.38	0.5	0.13	0.38	0.25	0.38	0.38	0.25	0.63	0.25	0.13
19																				0.13	0.13	0.38	0.25	0.25	0.5	0.25	0.5	0.25	0.13	0.25	0.38	0.25	0.38	0.25	0.38	0.38	0.38	0.63	0.25	0.13
20																					0.5	0.25	0.38	0.63	0.13	0.13	0.38	0.5	0.13	0.25	0.13	0.25	0.25	0.38	0.38	0.5	0.13	0.25	0.13	0.13
21																						0.25	0.38	0.38	0.13	0.38	0.13	0.5	0	0.13	0	0	0.5	0.38	0.38	0.5	0.38	0.25	0.13	0
22																							0.25	0.25	0.63	0.38	0.38	0.25	0.13	0.13	0.25	0.25	0.38	0.13	0.5	0.38	0.38	0.38	0.25	0.13
23																								0.25	0.38	0.13	0.25	0.38	0.13	0.38	0.25	0.13	0.25	0.38	0.38	0.38	0.25	0.25	0.13	0
24																									0.13	0.38	0.5	0.5	0.13	0.38	0.13	0.13	0.38	0.25	0.5	0.88	0.13	0.5	0.13	0
25																										0.13	0.38	0.25	0.25	0.25	0.38	0.38	0.25	0.13	0.25	0.25	0.5	0.25	0.38	0.13
26																											0.38	0.13	0	0	0	0	0.38	0.25	0.38	0.5	0.25	0.38	0	0
27																												0.13	0.13	0.13	0.25	0.25	0.38	0	0.38	0.38	0.13	0.38	0.25	0.13
28																													0.13	0.38	0.13	0.13	0.38	0.38	0.38	0.63	0.25	0.38	0.13	0.25
29																														0.25	0.38	0.5	0	0.13	0.25	0.13	0.25	0.25	0.63	0.25
30																															0.63	0.38	0.25	0.13	0.25	0.38	0.38	0.38	0.25	0
31																																0.38	0.25	0	0.13	0.13	0.38	0.38	0.63	0.25
32																																	0.13	0.13	0.13	0.13	0.38	0.13	0.5	0.25
33																																		0.25	0.5	0.5	0.38	0.38	0.13	0.13
34																																			0.25	0.38	0.38	0.13	0.13	0.25
35																																				0.63	0.25	0.5	0.25	0.13
36																																					0.25	0.63	0.13	0
37																																						0.25	0.38	0.25
38																																							0.25	0
39																																								0.5