PRPC: An Improved Density-based Projected Clustering Algorithm

Ye-In Chang, Chen-Chang Wu and Tsun-Kuei Huang
Dept. of Computer Science and Engineering,
National Sun Yat-Sen University,
Kaohsiung, Taiwan, Republic of China
Email: changyi@cse.nsysu.edu.tw

***Abstract*—Traditional clustering algorithms consider all of the dimensions of an input dataset in an attempt to learn as much as possible about each object described. In the high dimensional data, however, many of the dimensions are often irrelevant. Therefore, projected clustering is proposed. The DOC algorithm is one of well-known density-based algorithms for projected clustering. The FPC algorithm is an extended version of the DOC algorithm, it uses the mining large itemsets approach to find the dimensions of projected cluster. Although the FPC algorithm has used the technique of mining large itemsets to speed up finding projected clusters, it still needs many user-specified parameters to work. Moreover, the FPC algorithm applies a random approach for several times to get the medoid, which takes long time and may still find a bad medoid. Furthermore, the way to calculate the quality of a cluster can be considered in more details, if we take the weight of dimensions into consideration. In this paper, we propose an algorithm, PRPC (Parameter-Relationship-based approach for Projected Clustering), which improves those disadvantages. From our simulation results, we show that our algorithm is better than the FPC algorithm, in term of the execution time and the quality of clustering.**

***Index Terms*—Clustering, Data Mining, Density-based, Large Itemset, Projected Clustering.**

I. Introduction

The clustering problem has been discussed extensively in the database literature as a tool for similarity search, customer segmentation, pattern recognition, trend analysis, classification, bioinformatics [7], [17], and so on. In spatial data mining, clustering is a useful technique for discovering interesting data distributions and patterns in the underlying data. Given a large set of multidimensional data points, the data space is usually not uniformly occupied by the data points. Data clustering identifies the sparse and the crowded places, and hence discovers the overall distribution patterns of the data set [8].

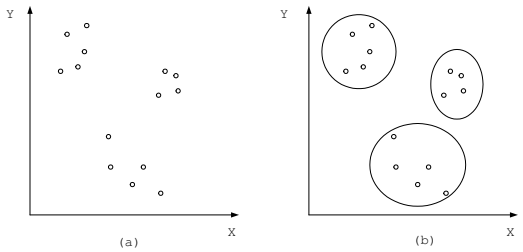

Fig. 1. The object of clustering

The problem of clustering can be defined formally as follows: given n data points in a d-dimensional metric space, partition the data points into k clusters such that the data points within a cluster are more similar to each other than data points in different clusters [11]. An example of clustering is depicted in Figure 1. The input patterns are shown in Figure 1-(a), and the desired clusters are shown in Figure 1-(b). In general, the clustering algorithms can be classified into three approaches: *partition*, *hierarchical* and *density-based* approaches. For the partitioning approach, there are k-means [13], PAM [13], CLARA [13], and CLARANS [15]. For the hierarchical approach, there are HAC [18], BIRCH [26], ROCK [10], and CURE [11]. For the density-base approach, there are CLIQUE [4], CAST [7], DBSCAN [9], and CDC [27].

Technology advances have made data collection easier and faster, resulting in larger, more complex datasets with many objects and dimensions. quality and speed. Traditional clustering algorithms consider all of the dimensions of an input datasets

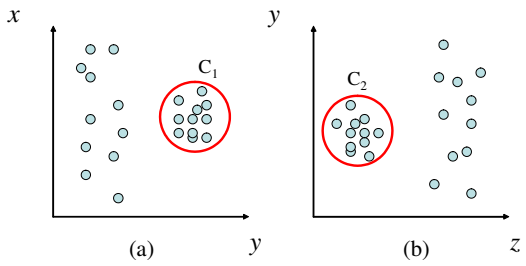

Fig. 2. Two clusters with different subspace: (a) cluster C_1 (x-y space); (b) cluster C_2 (y-z space).

in an attempt to learn as much as possible about each object described. In high dimensional data, however, many of the dimensions are often irrelevant. These irrelevant dimensions can confuse clustering algorithms by hiding clusters in noisy data. Feature selection methods have been employed somewhat successfully to improve cluster quality. These algorithms find a subset of dimensions on which to perform clustering by removing irrelevant and redundant dimensions. Unlike feature selection methods which examine the datasets as a whole, subspace clustering algorithms localize their search and are able to uncover clusters that exist in multiple, possibly overlapping subspaces.

Another reason that many clustering algorithms struggle with high dimensional data is the *curse of dimensionality*. As the number of dimensions in a datasets increases, distance measures become increasingly meaningless. Additional dimensions spread out the points until, they are almost equidistant from each other in very high dimensions.

Projected clustering algorithms are one answer to those challenges [1], [2], [3], [4], [6], [14], [16], [19], [20], [21], [22], [23], [24], [25]. A projected cluster is a subset C of data points together with a subset D of dimensions such that the points in C are closely clustered in the subspace of dimensions D. In Figure 2, two clusters exist in two different projected subspaces. Cluster C_1 exists in projected x–y space, and cluster C_2 exists in projected y–z space. We can divide the approaches of projected clustering into three classification: partitioning, density-based and hierarchical. In this paper, we focus on density-based projected clustering.

In density-based approach, users must give parameters α and ω. Parameter α is the density threshold of a cluster, *i.e*, the number of points in the cluster must be larger than this threshold. Parameter ω is the width of a cluster. The density-based approach does not need users to determining the number of clusters, and the resulting clusters are good with the high probability. The DOC algorithm [16] is the density-based clustering algorithm. The DOC algorithm develops a Monte Carlo algorithm for iteratively computing projected clusters, and the quality of cluster is good with the high probability. But, it still has some disadvantages, it needs long time to find all clusters in the data space, and it needs too many parameters. If parameters are not given advisably, the quality of the result will be seriously downgrade. The way of the DOC algorithm to find relevant dimensions is not efficient. It must execute long time to find the relevant dimensions, and the relevant dimensions that it finds is not always very suitable.

The FPC algorithm [24] is an extended version of the DOC algorithm, it uses the mining large itemsets approach to find the dimensions of projected cluster. It improves the way of selecting the dimensions of the DOC [16] algorithm. Finding the large itemsets is the main goal of mining association rules. The FPC algorithm finds the relationship between projected clustering and mining association rules, and translates the datasets into a transaction database with the medoid and the parameter ω. Each point and each dimension in the datasets are a transaction and an item in the transaction database, respectively. The items of a maximum large itemset are the related dimensions of the cluster, and the transaction that contains the maximum large itemset is the point of the cluster.

Although the FPC algorithm has used the technique of mining large itemsets to speed up finding projected clusters, it still needs three input parameters. Moreover, in the first step, to choose the medoid, the FPC algorithm applies a random approach for several times to get the medoid, which takes long time and may still find a bad medoid. Furthermore, the way to calculate the quality of a cluster can be considered in more details, if we take the weight of dimensions into consideration. Therefore, in this paper, we propose an algorithm, PRPC (Parameter-Relationship-based approach for

Projected Clustering), which improves those disadvantages.

The rest of this paper is organized as follows. In Section 2, we give a survey of some well-known projected clustering algorithms. In Section 3, we present a new projected clustering algorithm. In Section 4, we give a comparison of the performance of our new algorithm with the FPC algorithm. Finally, we give the conclusion and point out some future research directions in Section 5.

II. Related Works

In this section, we give a survey of some well-known projected clustering algorithms in recent years. First, we describe the DOC algorithm. Then, we describe the FPC algorithm that improves the DOC algorithm.

A. DOC

In the density-based approach, DOC [16], each cluster is defined as a hypercube with width 2ω, where ω is a user parameter which defines the width of cluster of each dimension. The clusters are formed one after another. To find a cluster, a pivot point is randomly chosen as the cluster center and a small set of objects is randomly sampled to form a tentative cluster around the pivot point. A dimension is selected if and only if the distance between the projected values of every sample and the pivot point on the dimension is less than ω. The tentative cluster is thus bounded by a hypercube with width 2ω. All objects in the dataset falling into the hypercube are grouped to form a candidate cluster. Many random samples and pivot points are then tried to form more candidate clusters, and a specially designed function is used to evaluate the quality of them. The candidate cluster with the best evaluation score is accepted, and the whole process is repeated to find other clusters.

Let S be a collection of $|S|$ d-dimensional points $p = (p_1, p_2, ..., p_d)$ in R^d. A projected cluster in S is a pair (C, D), where C is a subset of points and D is a subset of dimensions. A projected cluster must be *dense*. Specifically, the distance between every two points p and q in C in every dimension $i \in D$ must be at most ω, where ω is a problem parameter define the width of cluster of each dimension. Moreover, the size $|C|$ of cluster C(*i.e*, number of points in C) should be at least $\alpha|S|$, where $0 \leq \alpha \leq 1$ is a user parameter. Formally:

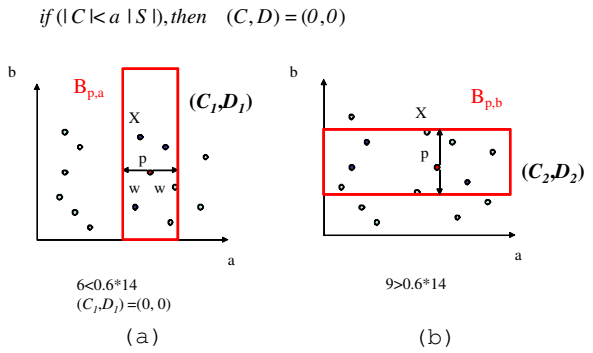

Fig. 3. A example of definition 1: (a) a cluster with the number of points which is less than dense threshold. (b) a cluster with number of points which is larger than dense threshold.

Definition 1: Let S be a set of points in R^d. Given a set of points $C \subseteq S$, a set of dimensions D, and parameters $0 < \alpha \leq 1$ and $\omega > 0$, (C, D) is a projected cluster if

1) $|C| \geq \alpha|S|$,
2) $\forall i \in D, max_{p \in C} p_i - min_{q \in C} q_i \leq \omega$, and
3) $\forall i \notin D, max_{p \in C} p_i - min_{q \in C} q_i > \omega$.

Figure 3 shown the example of definition 1. There are 14 points in dimension a and b. The α is 0.6 that decide by user. First, we pick X points to decide the dimension which the cluster correlates with. Then, we can draw the $B_{p,d}$. $B_{p,d}$ is the candidate cluster. Finally, we calculate the number of points in $B_{p,d}$. If the number of points in $B_{p,d}$ is less than $0.4 * 14$, the cluster will be discard and repeat these steps. In Figure 3-(a), the distance from point p to another point in $B_{p,a}$ on dimension a is less than ω. In Figure 3-(b), the distance from point p to another points in $B_{p,b}$ on dimension b is less than ω. In Figure 3-(a), the number of points in $B_{p,a}$ is less than $\alpha|S|$, so it can not be a cluster(*i.e*, $(C_1, D_1) = (0, 0)$). On the other hand, in Figure 3-(b), the number of points in $B_{p,b}$ is larger than ω, so it will be a cluster (C_2, D_2).

Definition 2: Let a be the number of points in a projected cluster C. Let b be the dimensionality of C. The quality of cluster C is defined by

$$\mu(a, b) = a \cdot (\tfrac{1}{\beta})^b$$

Large β favors large clusters with small subspaces over small ones of high-dimensionality and vice-versa. DOC uses this definition to estimate the

ID	a_1	a_2	a_3	a_4
1	1	2	3	8
2	2	1	9	6
3	3	2	6	3
4	4	8	1	2
5	9	6	2	1
6	7	3	3	2

Fig. 4. The dataset with four dimensions

ID	Itemset
1	$\{a_1, a_2\}$
2	$\{a_1, a_2\}$
3	$\{a_1, a_2, a_3, a_4\}$
4	$\{a_1, a_4\}$
5	$\{a_4\}$
6	$\{a_2, a_4\}$

Fig. 5. The transaction database

quality of cluster. It outputs a cluster that has the best quality every time.

B. Frequent-Pattern-based Clustering

FPC (Frequent-Pattern-based Clustering) [24] is the algorithm that improves the DOC [16], and first associates with the algorithm of mining large itemsets. It uses the miming associate rule to find the relevant dimensions of cluster. After finding the relevant dimensions, FPC uses the algorithm of DOC to find cluster. DOC discovers one cluster at a time, and FPC does too. Before using the mining associate rule to find the relevant subspace, we need to map the dataset to transaction database. The large itemset in the transaction database is equal to the subspace in dataset, and its support in the transaction database is equal to the size of cluster in this subspace. In Figure 4 is the original dataset, there are six points with four dimensions. FPC chooses the medoid randomly, and uses this medoid to create the transaction database. The point with ID = 3 is chosen as a medoid randomly, and the ω is 2 that decided by users. If the distance between points and medoid with this dimension is less or equal than 2 ($\omega = 2$), this dimension is chosen as an item of itemset. The result is shown as Figure 5.

After finding the maximum large itemsets, we can find two clusters with $\{a_1, a_2\}$ and $\{a_4\}$. The result of FPC is shown in Figure 6. Then, we input the result into refinement step, and output the best one to users.

ID	Itemset
1	$\{a_1, a_2\}$
2	$\{a_1, a_2\}$
3	$\{a_1, a_2, a_3, a_4\}$

(a)

ID	Itemset
4	$\{a_1, a_4\}$
5	$\{a_4\}$
6	$\{a_2, a_4\}$

(b)

Fig. 6. The results: (a) C_1 ($LBest$ is $\{a_1, a_2\}$) ; (b) C_2 ($LBest$ is $\{a_4\}$).

	C_1	C_2
R	1.5	1.58
μ	75	15

Fig. 7. The refinement step

In the refinement step, there are two formulas for measure the quality of clusters.

1) $\mu(a, b) = a \cdot (\frac{1}{\beta})^b$,
2) $R(C, D) = \frac{\sum_{x \in C} dist_D(x, \overline{X}(C))}{|C|}$

Formula 1 is from [16], and formula 2 is from [2]. Let a be the number of points in a cluster C. Let b be the number of dimensions in cluster C. $\overline{X}(C)$ is the centroid of cluster C, defined by $\overline{X}(C)_i = \frac{\sum_{x \in C} x_i}{|C|}$ for each dimension i. In order to dealing with clusters that have different number of dimensions fair, we use Manhattan segmental distance. Given two points p and q, their Manhattan segmental distance in subspace D is defined as $dist_D(p, q) = \frac{\sum_{i \in D} |p_i - q_i|}{|D|}$, where $|D|$ is the number of dimensions in D. β is a user parameter that indicates the importance of between points and dimensions. Formula 1 is a quality measure formula, and FPC uses it to measure the quality of a cluster. Formula 2 is a spread measure formula that is used to determine the compactness of a cluster. The best cluster has high quality and small spread. After calculating the μ and $R(C, D)$, the result will be ranking. The result is shown in Figure 7. In Figure 8, the sum of ranking of μ and $R(C, D)$ is be considered to select to the best cluster. The best cluster that has the minimum sum will be output.

III. A Parameter-Relationship-based Approach

In this section, we present our parameter-relationship-based approach for projected cluster-

	C_1	C_2
R	1	2
μ	1	2
Sum	2	4

Fig. 8. The ranking table

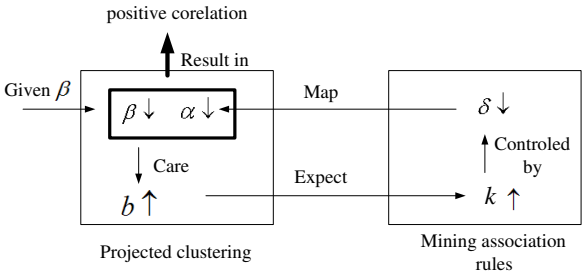

Fig. 9. The relationship between parameters α and β (positive corelation)

ing(PRPC). Our algorithm improves some problems of the FPC algorithm, and it is more efficient than the FPC algorithm. First, we find the relationship between parameters α and β, and decrease the number of required parameters. Next, we improve the way to find the medoid. These improvements let our algorithm cost less time to generate the best cluster than the FPC algorithm, and decrease the difficulty of deciding or choosing parameters. Then, we propose a measuring formula of quality in the refinement step. This formula makes our algorithm provide hight accuracy.

A. The Relationship Between Parameters

Parameter β is needed in the formula of quality, $\mu(a,b) = a \cdot (\frac{1}{\beta})^b$, where a is the number of points in a cluster, and b is the number of dimensions in a cluster, in the FPC algorithm [24]. If a cluster has many points and many dimensions, the value of its quality is high. But it is always a trade-off between the number of points in a cluster and the number of dimensions in a cluster. We can not have many points and many dimensions at the same time. Therefore, we need a parameter β to decide whether users favor the number of points in a cluster or the number of dimensions in a cluster. The value of β is larger than 0 and smaller than 1. If users favor the number of points, they will let $\beta \simeq 1$. In this way, based on the FPC algorithm, it will increase the number of points, and decrease the number of dimensions. On the other hand, if users favor the number of dimensions, they will let $\beta \simeq 0$. In this way, in order to increase the number of dimensions, the number of points will decrease.

As stated in [24], we can translate the problem of projected clustering into the problem of mining association rules with these relationships. These relationships make us have a different way to develop an efficient algorithm for projected clustering.

To reduce the number of required parameters, we observe some relationships between those parameters used in projected clustering and mining association rules. Figure 9 shows the relationship between parameters β, a, b, and α in projected clustering and parameters $|L_k|$, k and δ in mining association rules. As β decreases, it means that the users favor the large number of dimensions in a cluster. b is the number of dimensions in a cluster. As b increases, the number of items in the maximum large itemset (*i.e.*, k) will increase. If we want to have a large k, we must use the small value of the minimum support of the large itemset (*i.e.*, δ). The small value of δ means that we have a few points in a cluster (*i.e.*, α). A few points in a cluster means that we have small value of α. Therefore, through this derivation, we observe that, in fact, the role of parameter α is positive corelation to the role of β.

Moreover, we can find the similar relationship when β increases. Based on this observation between parameters α and β, α and β are positive corelation, we can conclude that we can decide the value of α according to the value of β. It improves the difficulty of users to decide the appropriate value of α. The users only need to decide the value of β. The value of β shows that users favor the number of points in a cluster or the number of dimensions in a cluster. As the value of β is decided, we can decide the value of α at the same time.

B. The Proposed Algorithm

Figure 10 shows our algorithm, and the variables used in our algorithm are listed in Table I. We use a two dimensions table as our input. Each row of the table is a point in the dataset, and each column of the table is a dimension in the dataset. Our output is the sets of clusters and dimensions. Each cluster has its own set of dimensions, respectively. To illustrate our

TABLE I
DESCRIPTION OF PARAMETERS

S_0	The original dataset
$\|S_0\|$	The size of the original dataset
$\|d_0\|$	The size of the original space
β	The importance of the size of the subspace over the size of the cluster
2ω	The width of a projected cluster in each dimension
δ	The minimum support of a large itemset
BM	The best medoid which we found in S_0
$LLarg$	The complete set of large itemsets that FP–growth found
$LBest$	The dimensions of the best cluster
$CBest$	The best cluster
$Medianall$	The median for all dimensions
TB	The table that we use to store the points in each dimension
TDB	The transaction database
C	The set of all clusters
D	The set of dimensions for each cluster
k	The number of clusters that we want
$TimeIJ$	The times that dimension j appears in a cluster i
$NumberI$	The number of dimensions that correlated with cluster i
μ^*	Our weight refinement formula

```
1:  Procedure PRPC(TB, β, TDB, ω, k);
2:  begin
3:      C := φ;
4:      D := φ;
5:      repeat
6:          SelectMedian(TB);
7:          FiltertoTDB(TB, ω);
8:          FindCluster(S0, β, TDB);
9:          C := C ∪ CBest;
10:         D := D ∪ LBest;
11:     until no cluster can be found;
12:     WRefinement(C, D, k);
13: end;
```

Fig. 10. Procedure $PRPC$

algorithm, we use an example, Example 2 shown in Figure 12, to explain our algorithm in each step. Our algorithm contains the following four steps.

Step 1: Find the medoid of the dataset. Before deciding the medoid, we have to find the median of the dataset. A median is the middle value in a set of numbers arranged in an increasing order. If there is an even number of values, the median is the average of the middle two values. For example: The median of the set $\{10, 12, 14, 19, 20\}$ is 14. The

```
1:  Procedure SelectMedoid(TB);
2:  begin
3:      BM := φ;
4:      dist[|S0|] := φ;
5:      find the Medianall of dataset S0;
6:      for each point i in each dimension j do
7:          dist[i] := dist[i] + |TB[i][j] − Medianall_j|;
8:      find the best medoid BM that has the minimum
9:      distance to point Medianall;
10: end;
```

Fig. 11. Procedure $SelectMedoid$

median of the set $\{2, 3, 4, 6, 8, 9\}$ is 5, which is the average of 4 and 6. After finding the median, we calculate the all distance between the median and other points. In order to decrease the execution time and the affection of different dimensions, we use the Manhattan distance in the distance calculation. The Manhattan distance between two points $x_1 = (x_{1,1}, ..., x_{1,d})$ and $x_2 = (x_{2,1}, ..., x_{2,d})$ is given by $d(x_1, x_2) = \sum_{i=1}^{d} |x_{1,i} - x_{2,i}|$. Then, we choose the point that has the minimum distance to the median as the medoid. The complete algorithm of this step is shown in Figure 11.

In Example 2, there are ten points with four dimensions in the dataset. We will find some clusters in this dataset by using our algorithm. We let parameter $\omega = 2$, and $\beta = 0.5$. These two parameters are decided by the user. Then, we find the median of the dataset of each dimensions. In our example, the median of the dataset is (4, 5, 3, 5). Next, we calculate the distances between the median and each point of the dataset. The $diff.$ in Figure 12 means the distance between the median and this point. We select the point with the minimum $diff.$ as the medoid in our dataset. The medoid BM is the point with ID = 7.

Step 2: Filter the points that the distance to the medoid is larger than ω, and translate the dataset to the transaction database. For example, if the distance between point A and the medoid with dimension a_1 is less than or equal to ω, we select a_1 as an item of the itemset of point A (*i.e,* translate). In this step, we translate the problem of projected clustering into the problem of mining association rules. The complete algorithm of this step is shown in Figure 13.

In Example 2, after finding the medoid BM (ID

ID	a_1	a_2	a_3	a_4	diff.
1	5	5	7	4	6
2	3	7	0	3	8
3	3	6	3	6	3
4	2	9	5	7	10
5	6	2	1	5	7
6	5	5	0	5	4
7	4	3	3	5	2
8	4	10	1	7	9
9	3	3	7	2	10
10	4	2	7	1	11

Fig. 12. Example 2: The dataset with four dimensions (BM=7)

```
1:  Procedure FiltertoTDB(TB, ω);
2:  begin
3:      for i := 1 to |S0|
4:          for j := 1 to |d0|
5:              if |TB[i][j] − BMj| ≤ ω
6:                  then TDB[i] := TDB[i] + j;
7:  end;
```

Fig. 13. Procedure $FiltertoTDB$

= 7), we will translate the dataset to the transaction database TDB with BM and ω (= 2). If the distance between the BM and this points with dimension a_1 is less than or equal to 2 ($\omega = 2$), dimension a_1 will be selected to itemset of this point. For example, the distances between the point with ID = 1 and the BM with dimension a_1, a_2, a_4 are 1 (5 - 4 = 1), 2 (5 - 3 = 2), 1 (5 - 4 = 1). These distances 1, 2, 1 is not larger than 2, so we select dimension a_1, a_2, a_4 as the items of itemset 1.

Step 3: Find the maximum large itemset, and then find the best cluster with the maximum large itemset. There are several different ways to find

ID	Itemset
1	$\{a_1, a_2, a_4\}$
2	$\{a_1, a_4\}$
3	$\{a_1, a_3, a_4\}$
4	$\{a_1, a_3, a_4\}$
5	$\{a_1, a_2, a_3, a_4\}$
6	$\{a_1, a_2, a_4\}$
7	$\{a_1, a_2, a_3, a_4\}$
8	$\{a_1, a_3, a_4\}$
9	$\{a_1, a_2\}$
10	$\{a_1, a_2\}$

Fig. 14. The TDB of Example 2 of PRPC ($\omega = 2$)

```
1:  Procedure FindCluster(S0, β, TDB);
2:  begin
3:      LBest[|d0|] := φ;
4:      CBest := φ;
5:      δ := β · |S0|;
6:      Construct the FP-tree from TDB with the minimum
7:      support δ;
8:      repeat
9:          If the FP-tree contains a single path P then
10:             for each combination β of the nodes
11:             in the path P do
12:                 generate pattern β ∪ α with support δ = the
13:                 minimum support of nodes in β;
14:         else
15:             for each ai in the header of the FP-tree do
16:                 begin
17:                     generate pattern β = ai ∪ α with support
18:                     ai.support;
19:                     construct β's conditional pattern base
20:                     and then β's conditional
21:                     FP-tree Treeβ;
22:                 end;
23:     until Treeβ = φ;
24:         LBest := max{LLarg};
25:         for i := 1 to |S0|
26:             if LBest ⊆ TDB[i]
27:                 CBest := CBest ∪ {xi};
28: end;
```

Fig. 15. Procedure $FindCluster$

the maximum large itemset for mining association rules. The main goal of mining association rules is to find the large itemsets, where a large itemset is a combination of items whose appearing times in the dataset is greater than a given threshold. In traditional methods of mining association rules, *e.g.*, the Apriori algorithm [5], all large itemsets will be kept. The maximum large itemset is the large itemset that has the maximum item in it. The FP–growth algorithm [12] is faster and more efficient than the Apriori algorithm. Therefore, we use it to find the maximum large itemsets. After finding the maximum large itemset, we find a cluster with the large itemset. The maximum large itemset is $LBest$ in our algorithm. $LBest$ is the dimensions of the best cluster. If we have the dimensions, we can find a cluster easily. If an itemset contains $LBest$, it is a point of a cluster with $LBest$. The complete algorithm of this step is shown in Figure 15.

In Example 2, we use the FP-growth algorithm to

ID	a_1	a_2	a_3	a_4
3	3	6	3	6
4	2	9	5	7
5	6	2	1	5
7	4	3	3	5
8	4	10	1	7

Fig. 16. The cluster with four dimensions at round 1

ID	a_1	a_2	a_3	a_4
1	5	5	7	4
2	3	7	0	3
6	5	5	0	5
9	3	3	7	2
10	4	2	7	2

Fig. 17. The dataset with four dimensions at round 2

mine the large itemset. We can find that the maximum large itemset is $\{a_1a_3a_4{:}5\}$. The α decides the number of points in a cluster and the minimum support value of the large itemset. We do not need to decide the value of α, it will be assign as β, so the value of α is 0.5. After finding the maximum large itemset $\{a_1a_3a_4{:}5\}$, we check the count of this maximum large itemset, if its count is less than the minimum support value δ, this itemset will be discarded. The count of $\{a_1a_3a_4{:}5\}$ is 5 equal to 5 (10 * 0.5), so it will be a $LBest$. Then, we find the best cluster with $LBest$. If an itemset contains $LBest$, it will be a point in a cluster. In our example, the points with ID = 3, 4, 5, 7, 8 contain $LBest$, so these points will be a cluster. Finally, we can find the best cluster which contains five points (IDs:3, 4, 5, 7, 8) with three dimensions (a_1, a_3, a_4). The result of round 1 is shown in Figure 16.

After round 1, the rest points is shown in Figure 17. We repeat the step 1 to step 3 until no cluster be found. At round 2, we can find that the maximum large itemset $LBest$ is $\{a_1, a_4\}$ and find the best cluster which contains five points (IDs:1, 2, 6, 9, 10) with two dimensions (a_1, a_4).

Step 4: Refine the result that we found. In this step, we use the following formulas:

1) $\mu(a, b) = a \cdot (\frac{1}{\beta})^b$,
2) $R(C, D) = \sum_{x \in C} dist_D(x, \overline{X}(C))/|C|$ and
3) $\mu^* = \sum w[i][j]/|D|$

Formulas 1 and 2 are from [24]. Let a be the number of points in a cluster C, and b be the number of dimensions in cluster C. $\overline{X}(C)$ is the centroid of cluster C, defined by $\overline{X}(C)_i = \sum_{x \in C} x_i/|C|$ for each dimension i. In order to deal with clusters that have different numbers of dimensions, we use the Manhattan segmental distance. Given two points p and q, their Manhattan segmental distance [2]in subspace D is defined as $dist_D(p, q) = \sum_{i \in D} |p_i - q_i|/|D|$, where $|D|$ is the number of dimensions in D. β is a user parameter that indicates the importance between points and dimensions. Formula 1 is a quality measuring formula [16]. We use it to measure the quality of a cluster. Formula 2 is a spread measuring formula [24] that is used to determine the compactness of a cluster. The best cluster has high quality and small spread. We design a new formula to measure the quality of a cluster. $w[i][j]$ is the weight of each dimension j in cluster i. Formula 3 is a correlation measurement of each dimension in a cluster. We give a weight to each dimension. If a dimension is an element in $LBest$, the weight of this dimension is assigned to 1. On the other hand, if a dimension is not an element in $LBest$, the weight of this dimension is assigned according to the times of appearance. As the times of appearance increases, the weight of this dimension decreases. If the number of dimensions that are not correlated with a cluster becomes small, the value of μ^* becomes large. We use these formulas in the refinement step.

In the refinement step, we calculate μ, R, and μ^* of each cluster that we find, and rank the results. The best cluster must has large μ, small R, and large μ^*. The cluster that has the minimum sum of these ranks is the best cluster. Then, clusters which have the smallest k scores are outputted, where k is the number of clusters that users want. The complete algorithms of this step and our refinement formula is shown in Figures 18 and 19.

In our Example 2, after all steps, we find two clusters in the datasets as shown in Figure 20, but we only need the best cluster. The refinement step is processed in this time. We use three formulas to measure the quality of clusters that we find. The weight of each dimensions of each cluster is shown in Figure 21. The result is shown in Figure 22. The sum of ranking score in shown in Figure 23. After ranking the quality of clusters, we select the best one to output.

```
Procedure WRefinement(C, D, k);
 begin
     Calculate R(C, D) for each cluster and ranking;
     Calculate μ(C, D) for each cluster and ranking;
     weightrefine(C, D);
     Add the ranking score of R(C, D), μ(C, D), μ*[i];
     Choose the the clusters which have the smallest
     k score;
 end
```

Fig. 18. Procedure $WRefinement$

```
Procedure weightrefine(C, D);
 /* timei is the times that dimension j appears in a cluster
     i */
 /* numberi is the number of dimensions that correlated
     with cluster i */
 begin
     for each dimension j of each cluster i in C
         If j ∈ LBest
             w[i][j] := 1;
         If j ∉ LBest
             w[i][j] := 1 - (TimeIJ/NumberI);
     for each dimension j of each cluster i in C
         μ*[i] := Σ w[i][j]/|D|;
 end
```

Fig. 19. Procedure $weightrefine$

ID	Itemset
3	$\{a_1, a_3, a_4\}$
4	$\{a_1, a_3, a_4\}$
5	$\{a_1, a_2, a_3, a_4\}$
7	$\{a_1, a_2, a_3, a_4\}$
8	$\{a_1, a_3, a_4\}$

(a)

ID	Itemset
1	$\{a_1, a_2, a_3, a_4\}$
2	$\{a_1, a_2, a_4\}$
6	$\{a_1, a_2, a_4\}$
9	$\{a_1, a_2, a_4\}$
10	$\{a_1, a_3, a_4\}$

(b)

Fig. 20. The results: (a) Cluster A ($LBest$ = $\{a_1, a_3, a_4\}$) ; (b) Cluster B ($LBest$ = $\{a_1, a_2\}$)

	a_1	a_2	a_3	a_4
Weight	1	$\frac{3}{5}$ $(1 - \frac{2}{5})$	1	1

(a)

	a_1	a_2	a_3	a_4
Weight	1	$\frac{1}{5}$ $(1 - \frac{4}{5})$	$\frac{3}{5}$ $(1 - \frac{2}{5})$	1

(b)

Fig. 21. The weights of each dimension: (a) Cluster A; (b) Cluster B.

	C_1	C_2
R	5.07	6.72
μ	40	20
μ^*	0.9 $(= \frac{18}{20})$	0.7 $(= \frac{14}{20})$

Fig. 22. The result of the WRefinement step

	C_1	C_2
R	1	2
μ	1	2
μ^*	1	2
Sum	3	6

Fig. 23. The ranking table

IV. Performance

In this section, we study the performance of the PRPC algorithm by simulation, and make a comparison with the FPC algorithm. The experiments were performed on the machine with a Intel Pentium 4 CPU, 1024MB of main memory, running Windows XP with Service Packet 2, and both algorithms were implemented in Java. Our experiments were run with synthetic datasets, and were repeated for 20 times to calculate the average execution time.

A. The Performance Model

The synthetic datasets used in our experiments were generated by randomly creating a number of synthetic clusters and their associated subspaces are the same as [24]. The range of points is [0, 100]. When generation cluster $i + 1$, about a half of its correlated dimensions are chosen from the associated dimensions of cluster i. This is intended to model the fact that different clusters often share some dimensions.

The number of dimensions associated with a cluster given by a Poisson random variable with mean μ, which has the additional restriction that this number must be at least 2 and at most d. Once the number of dimensions d_i for the cluster i is generated, the dimensions for each cluster are chosen using the following technique: the dimensions in the first cluster are chosen randomly. The dimensions for the ith cluster are generated by choosing $min\{d_{i-1}, d_i/2\}$ dimensions from the (i-1)th cluster and generating the other dimensions randomly. This iterative technique is intended to model the fact that different clusters frequently share subsets of correlated dimensions. The number of points in a cluater is $N_c = N \cdot (1 - outlier)$. Then, the number of points in cluster i is given by $N_c \cdot r_i / \sum_{i=1}^{k} r_i$, r_1, r_2, ..., r_k be the k random variables. We use these datasets to run our algorithm, and compare

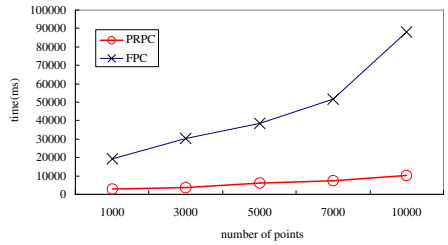

Fig. 24. A comparison of the execution time (α = 0.2 β = 0.2)

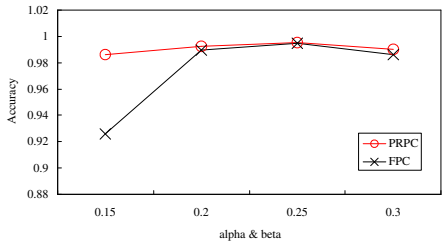

Fig. 25. A comparison of the accuracy

the execution time and accuracy with the FPC algorithm.

B. Experiment Results

First, we compare the execution time between our PRPC algorithm and the FPC algorithm. The values of α, β and ω is set to 0.2, 0.2 and 20. The number of points in the dataset is from 1000 to 10000, and the number of dimensions in the dataset is 20. We show that the execution time in our PRPC algorithm is shorter than that in the FPC algorithm shown in Figure 24. Moreover, as the number of points increases, both of the execution time of our algorithm and the FPC algorithm increase, because the time of finding maximum large itemset increases. Our PRPC algorithm chooses the medoid only once, so the execution time of our algorithm increases slowly. As the number of points increases, the difference between our PRPC algorithm and the FPC algorithm will increase.

Next, we show the comparison of the accuracy between our PRPC algorithm and the FPC algorithm in Figure 25. The detailed result is shown in Table II. The values of these parameters which we set are form 0.15 to 0.3. The number of points in this dataset is 10000, and the number of dimensions is 20. We show that the accuracy of our PRPC algorithm is higher than that of the FPC algorithm. When the value of parameter α is not given appropriately, the accuracy will be terrible. The reason is that some points of the cluster will be ignored. It is very important for the user to give an appropriate value of these parameters. When $\alpha = 0.15$, for example, the accuracy is low in both our PRPC algorithm and the FPC algorithm. However, our PRPC algorithm still has higher accuracy than the

TABLE II
A COMPARISON OF THE ACCURACY

α & β	PRPC	FPC
0.15	0.986	0.926
0.2	0.992	0.989
0.25	0.995	0.994
0.3	0.99	0.986

FPC algorithm. In this dataset, the appropriate value of parameter α is 0.25. When $\alpha = 0.25$, both of our PRPC algorithm and the FPC algorithm have high accuracy, However, the accuracy of our PRPC algorithm is higher than that of the FPC algorithm. The reason is that we use a weight refinement formula to refine the result of clustering.

Figure 26 show the comparison of the accuracy between our PRPC algorithm and the FPC algorithm with outliers. The value of these parameters which we set is 0.2. The number of points in this dataset is 10000, and the number of dimensions is 20. The percentage of outliers which we set is from 5% to 25%. We show that the accuracy of our PRPC algorithm and the FPC algorithm. As the number of the outliers increases, the accuracy will decrease. The reason is that the outliers will affect the choosing of the medoid. If the number of outliers is too large, the median of our PRPC algorithm will be affected. If the median is not decided appropriately, the medoid that we choose will be bad. The medoid will affect the accuracy of the results. In the FPC algorithm, if the number of outliers is too large, the probability of choosing the outliers as the medoid will be high. If the bad medoid is decided, the accuracy will be affected. The accuracy of our PRPC algorithm is always higher than that in the FPC algorithm,

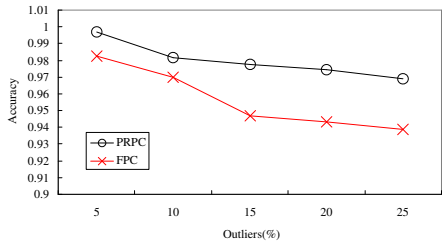

Fig. 26. A comparison of the accuracy with outliers

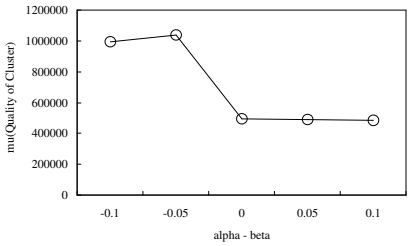

Fig. 27. The quality of clusters with different differences between α and β

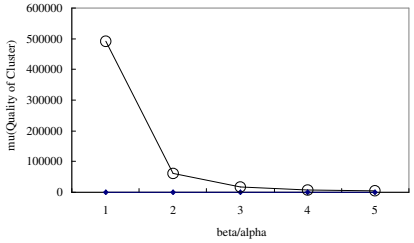

Fig. 28. The quality of clusters with different ratios between α and β

because we have an efficient way to choose the medoid, the probability of choosing the outliers as the medoid will be low.

Since there exists a relationship between parameters α and β according to our observation, we try to adjust the difference between parameters α and β to examine the quality of the results of clustering in our algorithm. The quality measure formula, $\mu(a,b) = a \cdot (\frac{1}{\beta})^b$ is used to measure the quality of the results. The formula considers the number of points and the number of dimensions in a cluster, and gives the different weight to each other. It is be used to measure the quality of a cluster in the density-based approach for projected clustering. We use this formula to measure our results of the clustering. Figure 27 shows our experiment results. When the difference between α and β (*i.e.,* α-β) is -0.05, the quality is the highest one. The value of α is between 0.1 and 0.3. If the value of α is too large, it is not reasonable and there will be no cluster outputted. It means that when the value of α is smaller than the value of β and the difference between these two parameters is small, the quality of the result of the clustering is the best. When the value of α is larger than the value of β, or the value of α is equal to the value of β, the quality of the clustering decreases. This is because the value of α will affect the number of dimensions in a cluster. The number of dimensions plays an important role in the quality measure formula. When the difference is -0.1, there are some points which not found. So, the quality of case (α-β = -0.1) is little less than the quality of case (α-β = -0.05).

As mentioned above, we observe that when the value of α is less than the value of β, the quality will be higher than other situations. So, we further try to adjust the ratio of β to α. Figure 28 shows our experiment results. As the ratio increases, the value of β increases, and the quality of cluster decreases. Because the value of β will affect the weight of the number of dimensions, a large value of β will result in the weight of the number of dimensions being decreased. According to these two experiment results, we observe that if the value of α is less than the value of β ($\alpha < \beta$) and the ratio of β to α is close to 1 ($\frac{\beta}{\alpha} \simeq 1$), the quality of the results of clustering is high.

V. Conclusion

Projected clustering has become more and more important, since it can discover interesting relationships and characteristics in high dimension datasets. An important problem of projected clustering is how to find the relative dimensions of each cluster. In this paper, we have proposed an efficient projected clustering algorithm to find the relative dimensions of each cluster with the mining frequent itemsets algorithm, since it improves the way of finding the medoid of the datasets. We also proposes a new

quality measure formula to improve the quality of the result of clustering. By our simulation result, we have shown that the execution time of our proposed algorithm is shorter than the FPC algorithm. Furthermore, we have shown that our algorithm has higher quality of clustering than the FPC algorithm. Moreover, our algorithm needs only two parameters, which makes our algorithm efficient.

A. Acknowledgements

This research was supported in part by the National Science Council of Republic of China under Grant No. NSC-99-2221-E-110-080-MY3 and National Sun Yat-Sen University.

REFERENCES

[1] C. C. Aggarwal, "A Human-Computer Cooperative System for Effective High Dimensional Clustering," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 221–226, 2003.

[2] C. C. Aggarwal, C. Procopiuc, J. L. Wolf, and J. S. Park, "Fast Algorithm for Projected Clustering," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 61–72, 1999.

[3] C. C. Aggarwal and P. S. Yu, "Finding Generalized Projected Clusters in High Dimensional Space," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 70–81, 2000.

[4] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 94–105, 1998.

[5] R. Agrawal and R. Srikant, "Fast Algorithm for Mining Association Rule," *Proc. of the 20th Int. Conf. on Very Large Database.*, pp. 487–499, 1994.

[6] J. An, J. X. Yu, C. A. Ratanamahatana, and Y. P. Chen, "A Dimensionality Reduction Algorithm and Its Application for Interactive Visualization," *Journal of Visual Languages and Computing*, Vol. 18, No. 1, pp. 48–70, Feb. 2007.

[7] A. Ben-Dor, R. Shamir, and Z. Yakhini, "Clustering Gene Expression Patterns," *Proc. of the 3rd Annual Int. Conf. on Computational Molecular Biology*, pp. 33–42, 1999.

[8] M. S. Chen, J. Han, and P. S. Yu, "Data Mining: An Overview from Database Perspective," *IEEE Trans. on Knowledge and Data Eng.*, Vol. 8, No. 6, pp. 866–883, Dec. 1996.

[9] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noises," *Proc. of the 2nd Int. Conf. on KDD*, pp. 226–231, 1996.

[10] S. Guha, R. Rastogi, and K. Shim, "ROCK: A Robust Clustering Algorithm for Categorical Attributes," *Proc. of Int. Conf. on Data Eng.*, pp. 512–521, 1999.

[11] S. Guha, R. Rastogi, and K. Shim, "CURE: An Efficient Clustering Algorithm for Large Databases," *Information Systems*, Vol. 26, No. 1, pp. 35–58, March 2001.

[12] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," *Proc. of the ACM SIGMOD*, pp. 57–87, 2000.

[13] L. Kaufman and P. J. Rousseeuw, "Finding Groups in Data: An Introduction to Cluster Analysis," *John Wiley, Sons, Inc.*, 1990.

[14] E. Ng, A. Fu, and R. Wong, "Projective Clustering by Histograms," *IEEE Trans. on Knowledge and Data Eng.*, Vol. 17, No. 3, pp. 369–383, March 2005.

[15] R. T. Ng and J. Han, "Efficent and Effective Clustering Methods for Spatial Data Mining," *Proc. of the 20th Int. Conf. on Vary Large Data Bases*, pp. 144–155, 1994.

[16] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali, "A Monte Carlo Algorithm for Fast Projective Clustering," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 418–427, 2002.

[17] L. Y. Tseng and S. B. Yang, "A Genetic Approach to the Automatic Clustering Problem," *Pattern Recognition*, Vol. 34, No. 2, pp. 415–424, Feb. 2001.

[18] E. M. Voorhees, "Implementing Agglomerative Hierarchical Clustering Algorithms for Use in Document Retrieval," *Proc. of Int. Conf. on Information Management*, pp. 465–476, 1986.

[19] K. G. Woo and J. H. Lee., "FINDIT:A Fast and Intelligent Subspace Clustering Algorithm Using Dimension Voting," *PhD Thesis, Korea Advance Institute of Science and Technology*, 2002.

[20] K. Y. Yip, D. W. Cheung, and M. K. Ng, "HARP:A Practical Projected Clustering Algorithm," *IEEE Trans. on Knowledge and Data Eng.*, Vol. 16, No. 11, pp. 1387–1397, Nov. 2004.

[21] K. Y. Yip, D. W. Cheung, and M. K. Ng, "On Discovery of Extremely Low-dimensional Clusters Using Semi-supervised Projected Clustering," *Proc. of the 21th Int. Conf. on Data Eng.*, pp. 329–340, 2005.

[22] M. L. Yiu and N. Mamoulis, "Clustering Gene Expression Data in SQL Using Locally Adaptive Metrics," *Proc. of ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pp. 35–41, 2003.

[23] M. L. Yiu and N. Mamoulis, "Frequent-Pattern Based Iterative Projected Clustering," *Proc. of the 3rd IEEE Int. Conf. on Data Mining*, pp. 689–692, 2003.

[24] M. L. Yiu and N. Mamoulis, "Iterative Projected Clustering by Subspace Mining," *IEEE Trans. on Knowledge and Data Eng.*, Vol. 17, No. 2, pp. 176–189, Feb. 2005.

[25] C. H. Yun, K. T. Chuang, and M. S. Chen, "An Efficient Method for Mining Market-Basket Clusters," *Information Systems*, Vol. 31, No. 3, pp. 170–186, May 2006.

[26] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: A Efficient Data Clustering Method for Very Large Databases," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 103–114, 1986.

[27] L. Zhao, J. Yang, and J. Fan, "A Fast Method of Coarse Density Clustering for Large Data Sets," *Proc. of Int. Conf. on Biomedical Engineering and Informatics*, pp. 1–5, 2009.