Journal of Southeast University (English Edition) Vol. 22, No. 4, pp. 490 – 495 Dec. 2006 ISSN 1003—7985

Mining maximal pattern-based subspace clusters in high dimensional space

Lu Yansheng Hu Rong Zou Lei Zhou Chong

(School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China)

Abstract: The problem of pattern-based subspace clustering, a special type of subspace clustering that uses pattern similarity as a measure of similarity, is studied. Unlike most traditional clustering algorithms that group the close values of objects in all the dimensions or a set of dimensions, clustering by pattern similarity shows an interesting pattern, where objects exhibit a coherent pattern of rise and fall in subspaces. A novel approach, named EMaPle to mine the maximal pattern-based subspace clusters, is designed. The EMaPle searches clusters only in the attribute enumeration spaces which are relatively few compared to the large number of row combinations in the typical datasets, and it exploits novel pruning techniques. EMaPle can find the clusters satisfying coherent constraints, size constraints and sign constraints neglected in MaPle. Both synthetic data sets and real data sets are used to evaluate EMaPle and demonstrate that it is more effective and scalable than MaPle.

Key words: subspace clustering; pattern similarity; maximal pattern-based subspace clusters

Cluster analysis seeks to make similar objects gather in the same cluster. The similarity between objects is often determined by distance measures over the full dimensions in the dataset, such as Euclidean distance, Manhattan distance, or cosine distance[1]. Since, in many applications, objects are expected to be correlated only under certain dimensions, subspace clustering has gained more popularity than full space clustering. Furthermore, clusters generated through subspace clustering may have different feature spaces.

The distance functions widely used in clustering are not always proper in capturing correlations among objects, because sometimes objects do not have approximate values even though strong correlations may still exist, while pattern similarity can capture the general tendencies of objects among subspaces in a high dimensional space. Therefore, a novel model, called Biclustering[2], based on pattern similarity was first proposed to solve this problem. In this paper, we explore a more effective and scalable approach to mine pattern-based subspace clusters.

In many applications, users are more interested in the objects that exhibit a consistent trend within the subset of dimensions, such as DNA microarray data analysis, automatic recommendation systems and target marketing systems. As a concrete example, in gene expression profiles analysis, it is particularly useful to identify the genes having similar trends across a set of conditions since the expression levels of some genes may be inflated or deflated systematically under some experiments. Thus, the absolute value is not as important as the trend. Such information is crucial in revealing the co-regulated genes in gene regulatory networks. The main contributions of this paper are as follows:

① We propose a novel pattern-based subspace clustering algorithm. For subspace clustering algorithms, objects and attributes are treated symmetrically, so that either objects or attributes can be treated as features. Typical datasets in science data and e-business have many more objects than attributes. Thus object-enumeration space is much larger than the attribute-enumeration space. We devise an algorithm that only exploits the attribute-enumeration space to conduct depth-first search and develop techniques to prune search subspaces.

② We consider the sign problem neglected in MaPle. So our algorithm identifies clusters satisfying coherent constraints, size constraints and sign constraints, which are of high biological significance.

③ We conduct experiments to validate the effectiveness of the proposed algorithm.

1 Related Work

Clustering techniques[3–5] often define the similarity between objects using distance measures over all dimensions. Ref. [6] presented that, for high dimensional spaces, a full-dimensional distance is often irrel-

Received 2006-03-01.

Foundation item: The National Natural Science Foundation of China (No. 60273075).

Biography: Lu Yansheng (1949—), male, professor, lys@mail.hust.edu.cn.

evant, as all of the objects are nearly equidistant from each other. Subspace clustering is a good solution for finding clusters in such data spaces by defining similarity on a selected subset of attributes for a set of objects. Several methods[7–10] have been introduced to discover interesting subspace clusters. None of the above algorithms satisfy our most important requirement: the ability to identify pattern similarity clusters in subspace. Since our work is highly related to pattern similarity clustering, we will describe it in more detail.

Cheng and Church[2] proposed biclustering, or simultaneous clustering on both genes and conditions, to discover knowledge from expression data. The problem of biclustering is to mine submatrices with low mean squared residue scores. A move-based algorithm[11] is a further improvement on the biclustering by allowing absent attribute values. The algorithms in Refs. [2, 11] adopt greedy search strategies, and thus may converge to a local minimum and cannot guarantee to find all clusters.

pCluster[12] can discover the entire set of clusters. It defines another similarity measure called pScore, which computes the change of values on the two attributes between the two objects. It first finds all column-pair and object-pair maximal dimension clusters called MDSs. Then it prunes object-pair MDSs and column-pair MDSs by turns until no pruning can be made. Finally, it inserts the remaining object-pair MDSs into a prefix tree. MaPle[13] studies the problem of maximal pattern-based clustering. It first computes and prunes attribute-pair MDSs and object-pair MDSs, then progressive refining and depth-first searching are devised for the maximal pattern-based clustering. However, the two ways need to maintain the set of MDSs for object-pair and attribute-pair simultaneously and thus consume a significant amount of time and space. Search space for object enumeration and pattern checking becomes huge when there are many MDSs.

How to find these clusters precisely and efficiently in considerable data space becomes a question worthy of consideration. For this reason, this paper proposes EMaPle, an effective maximal pattern-based mining algorithm, which mainly aims at reducing the unnecessary objects and sufficiently utilizing the constraints to compress search space. It runs faster than the previously developed pattern-based mining algorithms and decreases the magnitude of memory.

2 Problem Definitions

In this section, we introduce some important concepts that will be used throughout this paper.

To formulate how coherent the objects are on attributes, we introduce measure pScore.

Definition 1 (pScore)[13] Let $S = \{r_1, \ldots, r_n\}$ be a database with n objects. Each object has m attributes $A = \{a_1, \ldots, a_m\}$. We assume that each attribute is in the domain of real numbers. The value of object r_j on attribute a_i is denoted as $r_j.\ a_i$. For any objects $r_x, r_y \in S$ and any attributes $a_u, a_v \in A$, the pScore is defined as

$$\text{pScore}\left(\begin{bmatrix} r_x.\ a_u & r_x.\ a_v \\ r_y.\ a_u & r_y.\ a_v \end{bmatrix}\right) = \left\| (r_x.\ a_u - r_y.\ a_u) - (r_x.\ a_v - r_y.\ a_v) \right\|$$

Definition 2 (pattern-based cluster)[13] Let $R \subseteq S$ be a subset of objects in the database and $D \subseteq A$ be a subset of attributes. (R, D) is said a δ-pCluster if for any objects $r_x, r_y \in R$ and any attributes $a_u, a_v \in D$, $\text{pScore}\left(\begin{bmatrix} r_x.\ a_u & r_x.\ a_v \\ r_y.\ a_u & r_y.\ a_v \end{bmatrix}\right) \leqslant \delta$, where $\delta \geqslant 0$.

Definition 3 (maximal pCluster)[13] A δ-pCluster C is said maximal(or called a δ-MPC in short) if there exists no δ-pCluster C' such that C is a proper sub-cluster of C'.

A cluster with a small number of objects or a small number of attributes may be formed by chance. Users are not interested in these trivial and coincident clusters. As a result, users impose the minimum size threshold on objects and attributes.

Problem Statement Given a dataset D which contains values of objects on attributes, our task is to identify all MPCs with respect to a user specified threshold t, and we confine the minimum size constraints n_c(for attributes) and n_r(for objects) to mine significant clusters.

3 EMaPle Algorithm

We describe three steps of EMaPle (effective maximal pattern-based clustering) in this section. In the first step, we scan the dataset to find attribute-pair MDSs, maximal dimension sets containing only two attributes, and we propose the sign problem and solve it, which is introduced in section 3.1. In the second step, we use various pruning techniques to shrink data space. The pruning process is presented in section 3.2. We show how to find MPCs by a depth-first traversal with pruning in an attribute subset tree in section 3.3.

3.1 Finding attribute-pair MDSs

Unlike the method used in Ref. [12], we initialize end with $n_r - 1$ instead of 1 and set start no less than $n - n_r + 1$. The major ideas are illustrated in ex-

ample 1.

Example 1(finding attribute-pair MDSs) For 6 objects and 5 attributes in Fig. 1(a) with $t=1$, $n_r=3$, MDS for pair a_1, a_2 can be mined in the following steps.

Step 1 Compute and sort the discrepancies of the object values on attributes. Fig. 1(b) shows the result of this process.

Step 2 Run through the sorted list until discovering the first MDS. The two pointers start and end are set 0 and 2, respectively. The two values corresponding are -1 and 0. Since $0-(-1)\leqslant 1$, we move end forward until we find $2-(-1)>1$ and end $-$ start $\geqslant$ 2. Then $\{r_1, r_2, r_3, r_4, r_5\}$ is an MDS.

Step 3 Repeat the above process until start is $n-n_r-1$. We stop searching as start is greater than $6-3-1$. In total, we find one cluster $\{r_1, r_2, r_3, r_4, r_5\}$.

Attribute	r_1	r_2	r_3	r_4	r_5	r_6
a_1	5	4	5	7	-1	2
a_2	6	4	5	7	0	4
a_3	7	5	6	15	6	5
a_4	7	6	1	2	8	5
a_5	1	10	30	60	10	1

(a)

-1	-1	0	0	0	2
r_1	r_5	r_2	r_3	r_4	r_6

(b)

Fig. 1 A running example. (a) Raw database; (b) Finding MDSs for a_1, a_2

We notice that in DNA microarray, some expression levels are negative. A positive value means the gene i is over-expressed at sample j. A negative value implies the gene i is under-expressed at sample j. They have totally different biological means. Therefore, we cannot group them into the same cluster. Motivated by this, we need to validate the resulting MDS.

To solve the sign consistent constraint problem, we scan the resulting MDS to convert the expression values to 0 or 1. The convert function is as follows:

$$\text{sign}(x)=\begin{cases}1 & \text{if } x\geqslant 0\\ 0 & \text{if } x<0\end{cases}$$

We use bitmap to store the converted results. Then we process the bitwise NOT XOR operation. The attributes having 0 are removed. And the MDS having attributes less than n_c are deleted.

Consider the MDS $\{r_1, r_2, r_3, r_4, r_5\}$ above. It is a maximal size-constrained cluster, but it does not satisfy sign constraints. The sign-constrained cluster is $\{r_1, r_2, r_3, r_4\}$.

3.2 Global pruning

The results from the previous step is the collection of MDS. Pruning insignificant objects and attributes helps sharply reduce the data space. We utilize the information on the occurrences of objects and attributes in MDS. Our pruning lemma is stated as follows:

Lemma 1 (global pruning) Given the collection of MDS, we have the following conclusions: ① If attribute a_i appears in less than (n_c-1) MDS, then the MDS including a_i can be safely pruned; or ② If object r_j appears in less than (n_c-1) MDS containing a_i, then object r_j can be safely pruned; or ③ The MDS containing less than n_r objects can be removed.

Example 2 (pruning using lemma 1) Let us check the MDSs in Tab. 1. Attribute a_5 does not appear in any MDSs, therefore a_5 can be removed from the attribute set. Object r_5 can be removed in the same way. Moreover, object r_4 appears less than $(3-1)$, and can be pruned. Only attributes a_1, a_2, a_3, a_4 and objects r_1, r_2, r_3, r_6 survive after pruning.

Tab. 1 Global pruning

Objects	Attribute-pairs
$\{r_1, r_2, r_3, r_4\}$	$\{a_1, a_2\}$
$\{r_1, r_2, r_3\}$	$\{a_1, a_3\}$
$\{r_1, r_2, r_6\}$	$\{a_1, a_4\}$
$\{r_1, r_2, r_3, r_6\}$	$\{a_2, a_3\}$
$\{r_1, r_2, r_6\}$	$\{a_2, a_4\}$
$\{r_1, r_2, r_6\}$	$\{a_3, a_4\}$

After we get the initial MDSs, we can apply lemma 1 to prune objects and attributes. This process is carried out recursively until no pruning occurs. The complicated situation is that there can be more than one MDS for given attributes a_i and a_j. If an object occurs in a distinct MDS, we add the count for this object by 1, and the identical attribute-pair is contributed only once to the count. For example, there are two MDSs for $\{a_0, a_1\}$, that is $\{r_0, r_1, r_2\}$ and $\{r_1, r_2, r_3\}$, respectively. The occurrence for a_0 and a_1 is 1, and the occurrence for r_0, r_1, r_2, r_3 is also 1. The pruning algorithm is shown as follows:

Input: The set of MDSs, S; the minimum # of objects, n_r; the minimum # of attributes, n_c.

Output: All candidate attributes, Y'; the pruned MDSs, S'.

Methods: GlobalPrune (S, n_c, n_r).

For all the MDSs including a_i ($i=1$ to m) Do

Count the # of a_i ($i=1$ to m), Ca_i;

If $Ca_i < n_c-1$

Then return; //lemma 1①

Count the # of r_j ($j=1$ to n), Cr_j;

If $Cr_j < n_c-1$

Then prune r_j from S; //lemma 1②

If "prune r" cause the # of objects in MDSs $< n_r$

Then prune the MDSs; //Lemma 1③

Until no pruning occurs

As indicated in lemma 1, the candidate objects, attributes and MDSs are smaller than the original ones. In practice, the shrinking factor can be significant because in large datasets, many of the attributes and objects are often irrelevant and redundant and mask existing clusters in noisy data. The earlier we detect them, the better the algorithm performs.

3.3 Depth-first traversal with pruning in an attribute subset tree

In order to find the MPCs, we have to extend the sets of attributes in MDSs. The attributes surviving after pruning can be partitioned into exclusive small subsets. Assume that there is an order less than or equal to in the set of attributes. We organize these attribute power sets in a special kind of prefix tree, which not only allows us to store them efficiently, but also supports systematical enumeration of the possible combination of attributes. The structure of a complete attribute subset tree without pruning w. r. t. attribute $\{a_1, a_2, a_3, a_4\}$ in the running example is shown in Fig. 2.

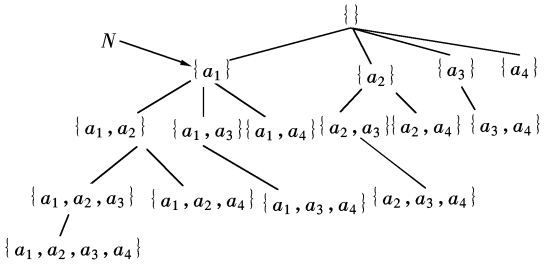

Fig. 2 Attribute subset tree for attribute $\{a_1, a_2, a_3, a_4\}$

Each node denotes an attribute or attribute subset and its corresponding maximal object set. The root of the tree is at the top level. Recursively, we can grow a node by adding one attribute to get the children nodes, and the children of the current node are arranged according to the attribute order. The tree is unbalanced since we deal with sets, not sequences. For instance, $\{a_1, a_2\}$ is the same as $\{a_2, a_1\}$. Therefore, these four attributes make up 15 attribute subsets.

Definition 4 (head attribute set and tail attribute set) The attribute set corresponding to each node will be referred to the head attribute set (HAS), and the possible extensions of the node constitute the tail attribute set (TAS).

Consider node N in Fig. 2, N's HAS is $\{a_1\}$ and the TAS is $\{a_2, a_3, a_4\}$. Because the dataset consists of a relatively small number of attributes, the tree can fit into memory.

By enumerating all the combinations of attributes, we ensure that the attributes in all MPCs are enumerated. It is obvious that a pure depth-first traversal is inefficient and pruning techniques must be introduced.

Assume that a_{i_m} is the last attribute in HAS, we can trim unpromising attributes from TAS that list before attribute a_{i_m}, or cannot form an MDS with any one attribute in HAS, or form an MPC with every attribute in HAS and the associated object set is not the subset of the object set in MPC for H. Thus, we have the following lemma.

Lemma 2 (attribute pruning) Given HAS $H = \{a_{i_1}, \ldots, a_{i_m}\}$, attribute a_j can be pruned from TAS if ① $j \leqslant i_m$; or ② there does not exist MDSc for pair (a_{i_k}, a_j), where $a_{i_k} \in H$; or ③ $C(XH)$ is MPC for attribute set H, and $C(X(H \cup \{a_j\}))$ is MPC for attribute set $H \cup \{a_j\}$, then we can move attribute a_j from TAS to HAS.

Example 3 (pruning using lemma 2③) In our running example. $(\{r_1, r_2, r_3, r_6\}, \{a_1, a_2\})$ is MDS for $\{a_1, a_2\}$, and $(\{r_1, r_2, r_3, r_6\}, \{a_1, a_2, a_3\})$ is MDS for $\{a_1, a_2, a_3\}$, thus node $\{a_1, a_2\}$ can be replaced by $\{a_1, a_2, a_3\}$.

Moreover, we can prune unnecessary subtrees that have not enough attributes, or have been absorbed by existing MPC, or the number of object subsets is less than n_r.

Lemma 3 (subtree pruning) Given node N, its HAS $H = \{a_{i_1}, \ldots, a_{i_m}\}$, and TAS $T = \{a_{j_1}, \ldots, a_{j_n}\}$, its corresponding object subset is X, the subtree of node N can be pruned if ① $|H \cup T| < n_c$; or ② for $X \subseteq X'$ and $H \subseteq Y$, there exists $C(XH)$ and $C(XY)$; or ③ $|X| < n_r$.

Based on the above pruning lemmas, the algorithm generating MPC is stated as follows:

```
Input: All candidate attributes, Y'; the pruned MDSc, S'.
Output: Maximal pattern-based cluster, MPC.
Method:
For i = 1 to n − n_c + 1 do
For j = i + 1 to n − n_c + 2 do
   Find MDS for pair(a_i, a_j); //level 2 of prefix tree
For each pair(a_i, a_j) do
   Call generate-MPC(current node C, MPC); //depth-first search
   Generate-MPC(current node C, MPC)
    Find C's TAS T; //lemma 2① and ②
    If |C∪T| < n_c
    Then stop searching and return; //lemma 3①
    Count the MDS for C;
    If the MDS for C is contained by some existing MPC or |X| <
n_r
    Then stop searching and return; //lemma 3② and ③
    Use lemma 2③ to prune T and get T';
    For each attribute a_j in T' do
      C_new = C∪{a_j};
```

Generate-MPC(C_new, MPC);
If C is a leaf and C's HAS is not in MPC
Then add C's HAS to MPC

4 Performance Experiments

All the experiments are conducted on a PC with an AMD Athlon 1700 + CPU and 512 MB memory running a Microsoft Windows XP operating system. We choose a bit vector representation for the object-pair MDS (MDSo in short) and attribute-pair MDS (MDSc in short). Each element in this vector has a 0-1 value corresponding to whether or not a given object is included in that cluster. We evaluate our EMaPle algorithm and MaPle with both synthetic and real life datasets. With synthetic data, we can embed clusters in specified subspaces. Since we know the locations, we can check whether the algorithm can recover all the clusters. For testing the performance of our algorithm, both synthetic and real life data are used. We present the experimental details in the following.

The synthetic datasets are generated from a synthetic data generator as in Ref. [12]. In order to test the scalability of our algorithm as we increase the objects in the data, we keep the number of dimension fixed at 30, set $t=1$, $n_c=8$, $n_r=0.01n$, the number n of objects varying from 3 000 to 7 000, and we embed 30 subspace clusters. The results are shown in Fig. 3. Since our subspace search strategy is not dependent on the number of objects but rather it searches in the dimension-enumeration space, the impact of the number of objects in EMaPle is not as large as that in MaPle.

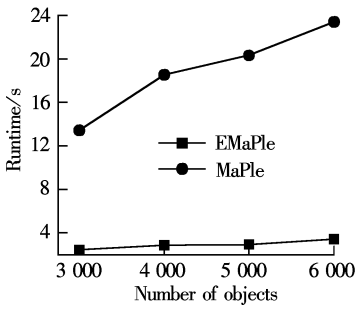

Fig. 3 Scalability with the number of objects

In the second scalability graph, shown in Fig. 4, we linearly increase the number of dimensions, set the size of object at 3 000, $t=1$, $n_c=0.2$ m, $n_r=30$, and we also embed 30 subspace clusters. Here the increase in runtime of EMaPle is obviously greater than in the previous case due to the increase in the number of attributes leading to the increase in the number of MDSc. And EMaPle runs also faster than MaPle.

Fig. 5 illustrates the results of comparing EMaPle to MaPle w. r. t. different minimal volumes of cluster

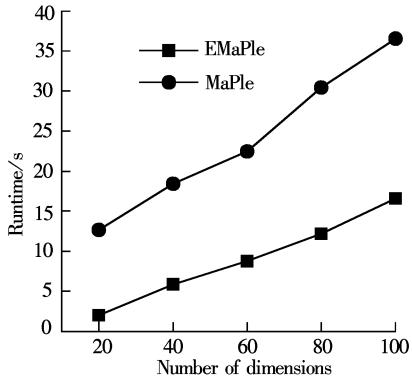

Fig. 4 Scalability with the number of dimensions

on the yeast microarray (available from http: //arep.med.harvard.edu/network_discovery). This benchmark dataset contains the expression level of 2 884 genes under 17 conditions. Because the number of MDSc decreases as the minimal number of objects increases, EMaPle runs faster on the larger minimal number of objects and outperforms MaPle.

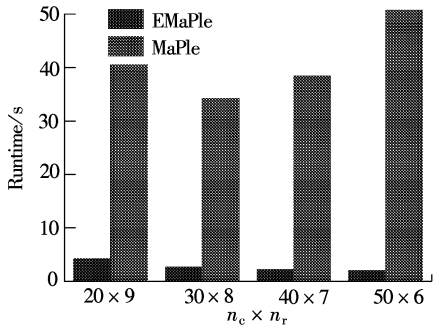

Fig. 5 Time comparison on different parameters

Fig. 6 shows the runtime spent on MDSo, MDSc and the whole EMaPle algorithm. The datasets used are synthetic datasets with $3\,000\times30$ and $3\,000\times100$ in size, respectively, and the yeast microarray, the size of which is $2\,884\times17$. We set the parameters of three datasets as follows: $t=1$, $n_c=6$, $n_r=27$; $t=1$, $n_c=8$, $n_r=27$; $t=2$, $n_c=7$, $n_r=40$. Our proposed solution takes advantage of the small number of dimensions

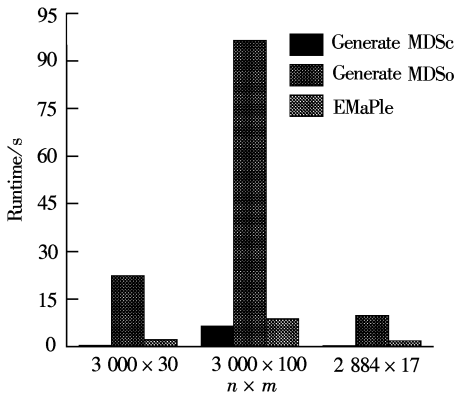

Fig. 6 Time comparison on MDSo, MDSc and EMaPle

compared to the number of objects by only calculating MDSc. Time spent on MDSo even exceeds that of EMaPle. Hence, our algorithm has better performance.

5 Conclusion

We are interested in automatically identifying projections of the input data to a subset of attributes. The solution we propose, EMaPle, has been designed to find clusters embedded in the subspaces of original data space. Experimental results demonstrate the effectiveness and efficiency of our method.

In the future, we hope to extend our algorithm to mining three dimensional clusters in gene-sample-time microarray data. It is very meaningful to investigate the pattern similarity in such data sets.

References

[1] Han J, Kamber M. *Data mining*: *concepts and techniques* [M]. San Francisco: Morgan Kaufmann, 2001.

[2] Cheng Y, Church G M. Biclustering of expression data [A]. In: *Proceedings of the* 8*th International Conference on Intelligent System for Molecular Biology*[C]. San Diego, CA, 2000. 93 - 103.

[3] Han J, Ng R T. Efficient and effective clustering method for spatial data mining [A]. In: *Proceedings of the* 8*th International Conference on Very Large Data Bases*[C]. Santiago, Chile, 1994. 144 - 155.

[4] Ester M, Kriegel H P, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise [A]. In: *Proceedings of the* 2*nd International Conference on Knowledge Discovery and Data Mining* [C]. Portland, Oregon, 1996. 226 - 231.

[5] Guha S, Rastogi R, Shim K. CURE: an efficient clustering algorithm for large databases. [A]. In: *Proceedings of ACM SIGMOD International Conference on Management of Data*[C]. Seattle, USA, 1998. 73 - 84.

[6] Beyer K, Goldstein J, Ramakrishnan R, et al. When is "nearest neighbor" meaningful [A]. In: *Proceedings of the* 7*th International Conference on Database Theory*[C]. Jerusalem, Israel, 1999. 217 - 235.

[7] Agrawal R, Gehrke J, Gunopulos D, et al. Automatic subspace clustering of high dimensional data for data mining applications[A]. In: *Proceedings of the ACM SIGMOD International Conference Management of Data*[C]. Seattle, USA, 1998. 94 - 105.

[8] Aggarwal C C, Procopiuc C, Wolf J L, et al. Fast algorithms for projected clustering [A]. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*[C]. Pennsylvania, USA, 1999. 61 - 72.

[9] Aggarwal C C, Yu P S. Finding generalized projected clusters in high dimensional spaces [A]. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*[C]. Dallas, Texas, 2000. 70 - 81.

[10] Goil S, Nagesh H, Choudhary A. Mafia: efficient and scalable subspace clustering for very large data sets[R]. Evanston: Northwestern University, 1999.

[11] Yang J, Wang W, Wang H, et al. Delta-clusters: capturing subspace correlation in a large data set [A]. In: *Proceedings of the* 18*th International Conference on Data Engineering* [C]. San Jose, CA, 2002. 517 - 528.

[12] Wang H, Wang W, Yang J, et al. Clustering by pattern similarity in large data sets [A]. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*[C]. Madison, Wisconsin, 2002. 394 - 405.

[13] Pei J, Zhang X, Cho M, et al. Maple: a fast algorithm for maximal pattern-based clustering [A]. In: *Proceedings of the Third IEEE International Conference on Data Mining*[C]. Florida, USA, 2003. 259 - 266.

高维空间基于样式相似性的最大子空间聚类

卢炎生　胡　蓉　邹　磊　周　翀

（华中科技大学计算机科学与技术学院，武汉 430074）

摘要：研究了基于样式相似性的子空间聚类问题，使用样式相似性作为相似性度量. 与在所有维或者子维集上聚集距离相近的对象的传统聚类方法不同的是，样式相似性寻找的是这样一种有趣的样式：对象在子维上呈现出相同起伏的一致变化. 提出了一种挖掘基于样式相似性的最大子空间聚类的方法 EMaPle. 一般情况下数据集属性数目远小于对象数目，因此仅在属性计数空间查找簇，然后运用一些修剪策略. 该方法能够找到同时满足一致性约束、大小约束和被 MaPle 忽视了的符号约束的聚类. 在合成和实际数据集上的实验结果表明该算法优于原来的 MaPle 算法.

关键词：子空间聚类；样式相似性；基于样式相似性的最大子空间聚类

中图分类号：TP311