Mining Relaxed Closed Subspace Clusters

Erich A. Peterson
University of Arkansas at Little Rock
2801 S. University Ave.
Little Rock, AR 72204
eapeterson@ualr.edu

Peiyi Tang
University of Arkansas at Little Rock
2801 S. University Ave.
Little Rock, AR 72204
pxtang@ualr.edu

ABSTRACT

This paper defines and discusses a new problem in the area of subspace clustering. It defines the problem of mining closed subspace clusters. This new concept allows for the culling of more high quality and less redundant clusters, than that of traditional clustering algorithms. In addition, our method contains a relaxation parameter, which allows for the classification of qualifying clusters into mutually exclusive bins of varying quality—extending the problem to mining relaxed closed subspace clusters. These concepts culminate in a new algorithm called Relaxed Closed Subspace Clustering (RCSC).

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*data mining*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Clustering*

General Terms

Algorithms, Theory.

Keywords

Closed subspace cluster, maximal subspace cluster, relaxed closed subspace cluster, relaxed interval, subspace clustering, data mining.

1. INTRODUCTION

We say *subspace clustering* [18, 2, 6, 9, 10, 3, 19, 1] (as opposed to classic clustering [4, 11, 12], pattern-based clustering [16, 13, 17, 22], or correlation clustering [13]) seeks to find clusters of objects that are "close" to one another on a certain subset of all the attributes within a database, where the "closeness" of the clusters is determined by some measure. The literature will sometimes refer to subspace clustering as "projection clustering" (which originally referred to a different subproblem, but has since blended into the same problem) [13]. Technically, and disseminated in [13], the problem of finding subspace clusters, should be referred to as "finding clusters in axis-parallel subspaces". In this paper, we will simply refer to it as subspace clustering. The invention of subspace clustering, was formulated to provide a more efficient way of mining clusters, when the dimensionality of the database is high. When dimensionality is high, nearly all objects are equidistant from one another—a phenomenon known as the *curse of dimensionality* [5]. Work has been done in transforming varying forms of the subspace clustering problem, into a certain kind of frequent itemset mining problem [23, 14].

In recent years, using what is known as the Galois (closure) operator, researchers have defined what is called a *closed itemset* and presented algorithms for the mining of frequent closed itemsets, both in static [15] and in evolving/streaming databases [8]. Moreover, Song et al. [21] have presented an algorithm that partitions the support threshold into intervals $(I_1, \ldots, I_n)$ known as *relaxed intervals* (based on a user-specified relaxation factor). Each closed itemset found is then assigned to an interval I_i based on its support. The formulation of a *relaxed closed itemset* can then be made; that is, a closed itemset X is considered relaxed and closed, if and only if, there exist no proper superset X' of X such that both belong to the same interval.

In this paper, we present the new problem of mining for *relaxed closed subspace clusters*, drawing inspiration from the aforementioned algorithms and concepts, and an algorithm called RCSC to mine for these newly defined clusters. This allows for the culling of more high quality and less redundant clusters, than that of traditional clustering algorithms. The rest of this paper is as follows: the definition of the problem is presented in Section 2; the algorithm to mine these newly defined clusters is shown in Section 3; some experimental results are shown in Section 4; finally, in Section 5 conclusions and future work are explained.

2. DEFINITION OF PROBLEM

Let O be a set of objects and A a set of attributes (where each attribute's domain is bounded) in a database S. If we have objects $x, y \in O$ and attributes $a, b \in A$, we denote the value of object x on attribute a as v_{xa} and the value of object y on attribute b as v_{yb}. Let R_a denote the range of attribute a, i.e., $\mid max_{x \in O}(v_{xa}) - min_{y \in O}(v_{ya}) \mid$. Thus, the *relative distance* between any two objects x and y on attribute a is $\mid v_{xa} - v_{ya} \mid / R_a$. Let T be a subset of objects from all objects O (i.e., $T \subseteq O$), and D be a subset of attributes from all attributes A (i.e., $D \subseteq A$). In addition, let the user-specified value δ, be known as the *relative distance threshold* $0 \leq \delta \leq 1$. This leads to the common definition of a *subspace cluster* in Definition 1.

Definition 1. (Subspace Cluster) Given a subset of all objects $T \subseteq O$ and a subset of all attributes $D \subseteq A$, (T, D) is considered a subspace cluster, if and only if, the relative distance between any objects in T, on all attributes in D, are less than δ .

In addition, given two subspace clusters (T, D) and (T', D'), if $T \subseteq T'$ and $D \subseteq D'$, we say that (T, D) is a *sub-cluster* of (T', D'), and conversely, that (T', D') is a *super-cluster* of (T, D). Moreover, if either $T \subset T'$ or $D \subset D'$ is true, then we say that (T, D) is a *proper sub-cluster* of (T', D'), and conversely, that (T', D') is a *proper super-cluster* of (T, D), denoted as $(T, D) \subset (T', D')$.

Subspace clusters exhibit the downward closure property. More precisely, given a subspace cluster (T, D), all sub-clusters of (T, D) are also subspace clusters. Therefore, once a cluster has been found, the reporting of its sub-clusters is redundant, and those clusters are called *redundant clusters*. In all but the most trivial databases, the mining of subspace clusters according to Definition 1, leads to the discovery of redundant subspace clusters. That is, given a subspace cluster (T', D'), there could be a large number of subspace clusters (T, D), that are sub-clusters of (T', D'). This has led to the development of algorithms such as ones that only mine *maximal subspace clusters* [14][1]. The definition of a maximal subspace cluster is given in Definition 2.

Definition 2. (Maximal Subspace Cluster) A subspace cluster (T, D) is maximal, if and only if, there does not exist a subspace cluster (T', D') such that $(T, D) \subset (T', D')$.

The net result is a reduction in the number of subspace clusters, but also leads to the loss of any measurement of quality of the clusters, that is, the relative closeness of the objects in the cluster. With traditional subspace clustering algorithms, as long as the cluster's objects relative distance is below the relative distance threshold δ they are reported.

This lack of a quantitative quality measurement inspired our new concept and definition of a *closed subspace cluster*. Given a subspace cluster (T, D), we use its *diameter* defined as:

$$\xi_D^T = max_{a \in D, x \in T, y \in T}(\mid v_{xa} - v_{ya} \mid / R_a) \qquad (1)$$

to define the qualify of the cluster. With this concept a closed subspace cluster is defined in Definition 3.

Definition 3. (Closed Subspace Cluster) A subspace cluster (T, D) is closed, if and only if, there does not exist a subspace cluster (T', D') such that $(T, D) \subset (T', D')$ and $\xi_D^T = \xi_{D'}^{T'}$.

The previous definition is the natural and intuitive extension of the concept of closure in other domains (e.g., itemset mining). It also allows for the discovery and evaluation of subspace clusters of differing diameters. However, tiny differences in diameter between two clusters $(T, D) \subset (T', D')$ can cause both to be closed. If many tiny differences in diameter exist between clusters with this type of relationship, too many will be reported. For example, if we have two subspace clusters $(T, D) = (\{o_1\}, \{a_1\})$ with $\xi_D^T = 0.45$, and $(T', D') = (\{o_1\}, \{a_1, a_2\})$ with $\xi_{D'}^{T'} = 0.44$, even though (T', D') is a super-cluster of (T, D), because the diameters are not equal, both could be closed subspace clusters. Faced with this problem we introduce a method to relax the diameters considered distinct.

Let the user-specified value p, $1 \leq p < \infty$, be known as the *relaxation factor*. It is used to divide the range of values between 0 and δ, into p discrete intervals I_i $(i = 1, \ldots, p)$ as:

$$I_i = [\frac{\delta}{p}(i-1), \frac{\delta}{p}i) \qquad (2)$$

Each subspace cluster can then be assigned to an interval I_i, based on the cluster's diameter ξ_D^T. For example, given $\delta = 0.5$ and $p = 4$, we would have the following intervals: $I_1 = [0, 0.125)$, $I_2 = [0.125, 0.25)$, $I_3 = [0.25, 0.375)$, and $I_4 = [0.375, 0.5)$. Given a subspace cluster (T, D) with $\xi_D^T = 0.2$, it would belong to interval I_2, since $0.2 \in I_2$. We denote the interval to which a subspace cluster (T, D) belongs to as I_D^T.

Finally, we are able to present the definition of a *relaxed closed subspace cluster*, which is presented in Definition 4.

Definition 4. (Relaxed Closed Subspace Cluster) A subspace cluster (T, D) is relaxed and closed, if and only if, there does not exist a subspace cluster (T', D') such that $(T, D) \subset (T', D')$ and $I_D^T = I_{D'}^{T'}$.

One can see, that as p is increased, so will be the number of relaxed and closed clusters reported. In fact, as p approaches ∞, the problem reduces to the classic subspace clustering problem (see Definition 1)—making it a special case of our algorithm. When $p = 1$, we have only one interval and the problem is reduced to the maximal subspace clustering problem (see Definition 2). Also, one can think of the relaxed and closed clusters found within lower numbered intervals as being of a higher quality. That is, their diameters are smaller and thus the objects within each cluster are more relatively close.

Comparing the definitions we have discussed on a simple set of subspace clusters will help illustrate their differences. Suppose we have four subspace clusters with respect to $\delta = 0.50$ as shown in Table 1. According to the definition of a sub-cluster, we have $C_1 \subset C_2 \subset C_3 \subset C_4$. Assume that $p = 2$. Table 2 shows the different subspace clusters found using each of the three aforementioned definitions.

The traditional subspace cluster definition would find all clusters, because in all clusters, all the objects that are in a cluster have a relative distance from each other which is less

Table 1: Example of Four Clusters ($\delta = 0.50$)

Cluster	**Diameter ξ**
$C_1 = (\{o_1\}, \{a_1\})$	0.20
$C_2 = (\{o_1, o_2\}, \{a_1\})$	0.20
$C_3 = (\{o_1, o_2\}, \{a_1, a_2\})$	0.26
$C_4 = (\{o_1, o_2\}, \{a_1, a_2, a_3\})$	0.30

[1]In [14], Liu et al. define them as maximal δ-nClusters, where the 'n' stands for neighbor.

Table 2: Subspace Clustering Methods

Method	Subspace Clusters Found
Traditional (Def. 1)	C_1, C_2, C_3, C_4
Maximal (Def. 2)	C_4
Closed (Def. 3)	C_2, C_3, C_4
Relaxed Closed (Def. 4)	$C_2 : I_1, C_4 : I_2$

than 0.50 on all attributes in the cluster. According to the maximal subspace cluster definition, only C_4 is a maximal subspace cluster, because C_1, C_2 and C_3 are sub-clusters of C_4. Using the closed subspace cluster definition, we would find that C_2, C_3, and C_4 are closed subspace clusters, because each of them has a distinct diameter and there is no supercluster with the same diameter. C_1 is not a closed subspace cluster because it has the same diameter as C_2 and is a proper sub-cluster of C_2. Finally, using the relaxed closed subspace cluster definition with $p = 2$, only C_2 and C_4 are relaxed and closed subspace clusters. This is because C_1 and C_2 belong to I_1 and C_3 and C_4 belong to I_2. But, $C_1 \subset C_2$ and $C_3 \subset C_4$; thus, C_1 and C_3 cannot be relaxed and closed subspace clusters according to Definition 4.

3. RELAXED CLOSED SUBSPACE CLUSTERING (RCSC)

Now that the problem definitions for closed subspace clusters and relaxed closed subspace clusters have been presented, we now turn to the dissemination of an algorithm for the mining of relaxed closed subspace clusters called RCSC.

Our strategy to find relaxed closed subspace clusters takes two steps: (1) we first find all the subspace clusters with respect to relative distance threshold δ (and a couple of other user-specified parameters to be discussed), and (2) find the relaxed closed subspace clusters for each interval by pruning away non-relaxed and non-closed subspace clusters. The first step is accomplished by transforming the problem into a quasi-frequent itemset mining problem.

3.1 Transformation to Itemset Mining Problem

As mentioned in Section 1, several algorithms have transformed varying forms of the subspace clustering problem, into a certain kind of itemset mining problem [23, 14]. We take inspiration from this idea (especially in [14]) in this subsection and use it as a part of our RCSC algorithm.

Let database S, shown in Table 3, be a running example with attributes a, b, and c and objects $0, 1, 2$, and 3. The range of values for each attribute in S is: $R_a = 4 - 1 = 3$, $R_b = 12 - 8 = 4$, and $R_c = 1.0 - 0.1 = 0.9$. Also, let $\delta = 0.51$.

Table 3: Sample Database S

	a	**b**	**c**
0	1	9	0.1
1	4	10	0.6
2	2	12	1.0
3	3	8	0.5

We start the process by making a projection of each of the attributes in S, and sort the values of each object on that attribute in ascending order. Table 4 shows these three projections made from the sample database S.

Table 4: Sample Database S

a

1	2	3	4

b

8	9	10	12

c

0.1	0.5	0.6	1.0

Next, two pointers s and t ($s < t$) are used to point to values in this projection. A search is then performed for a combination of positions for s and t, where the difference between the values t and s is less than $\delta \times R_a$, and either the difference between $(t + 1)$ and s, or t and $(s - 1)$ is greater than or equal $\delta \times R_a$. Once such a combination is found, the attribute a and the sorted list of the object indices represented between s and t are entered into a what is called a *maximal object list*. For example, for attribute b, if pointers s and t are pointed to values 8 and 10 respectively, then because $t - s < \delta \times R_b$—that is $10 - 8 < .51 \times 4$—and $((t + 1) - s) \geq \delta \times R_b$, b is entered in the maximal object list along with the object indices $0, 1, 3$ for values $9, 10, 8$ respectively (the object indices between s and t inclusive).

The maximal object list M constructed from our sample database is shown in Table 5.

Table 5: Maximal Object List M

Attribute	Maximal Object Set
a_0	$\{0, 2\}$
a_1	$\{2, 3\}$
a_2	$\{1, 3\}$
b_0	$\{0, 1, 3\}$
$*b_1$	$\{0, 1\}$
b_2	$\{1, 2\}$
c_0	$\{0, 3\}$
c_1	$\{1, 3\}$
c_2	$\{1, 2\}$

The result is a list, of the set of maximal objects, that are relatively close on a single attribute. Subscripts are used to distinguish which objects are relatively close on an attribute; the attribute and subscript together are called an *attribute symbol*. Without a subscript, we could not later tell that objects 0 and 3 are not relatively close on attribute a, even though objects 0 and 2 are, and 2 and 3 are. At this stage, it is not absolutely necessary to prune proper sub-clusters for the rest of the algorithm to preform accurately, but it decreases the size of the maximal object list and thus impacts performance. So, let us prune b_1 because it is a proper sub-cluster of b_0.

As can be seen in Table 5, there can be overlapping objects between two or more attribute symbols' maximal object sets. For example, both a_0 and a_1 have object 2 in their maximal object sets. The more overlap that is present, the larger the maximal object list M will be—and the larger the hit on performance will be. Again, as can be found in [14], a user-specified *overlapping threshold* ω is used to control the percentage of overlap tolerated. That is, if the number of objects in the most recently inserted maximal object contains k objects, then the next maximal object set inserted must have no more than $\omega \times k$ objects in common with it.

Now, the maximal object list is used to create what is known as an *attribute list*. For each object within the maximal object list, we make an entry into the attribute list and record next to it each attribute symbol that has that object in its maximal object set. For example, object 0 is in the maximal object set of attribute symbols a_0, b_0, and c_0 (excluding b_1, which we pruned). Table 6 shows the attribute list L formed from the maximal object list M in Table 5. (N.B. the transformation from M to L is a lossless one.)

Table 6: Attribute List L

Object	Attribute List
0	a_0, b_0, c_0
1	a_2, b_0, b_2, c_1, c_2
2	a_0, a_1, b_2, c_2
3	a_1, a_2, b_0, c_0, c_1

Once the attribute list L has been derived, the problem of mining subspace clusters (Definition 1) is reduced to a quasi-frequent itemset mining problem, with L representing our database to be mined. For clarity, let us briefly review frequent itemset mining. Frequent in the literature, when describing itemset mining, is its analogy to a shopping basket. Let A be some finite set of unique items, i.e. $A = \{a_1, \ldots, a_n\}$ $(0 \leq n < \infty)$. Any non-empty member α in the power set of A is called an itemset (or basket of items). Let a database D consist of transactions, with each transaction being associated with some non-empty itemset (basket). Given two itemsets α and β, if $\alpha \subseteq \beta$, we say β supports α. The number of transactions within D which support an itemset α is defined as $Supp_D(\alpha)$. Given a user-defined *support threshold* η, if $Supp_D(\alpha) \geq \eta$, then α is considered *frequent*. Thus, the goal of frequent itemset mining is to find all itemsets within D which are frequent.

Let L represent the transaction database D (described above); all unique attribute symbols represent the finite set of items A; each attribute list represent an itemset; and each object in L represent a transaction. Notice that for a given set of objects $T \subseteq O$, if the attribute lists of all the objects within T contain the same itemset α of attribute symbols, then T and the attributes D in α form a subspace cluster (T, D). This is because the relative distance between any objects in T on any attribute in D is less than relative distance threshold δ (see Definition 1). To find the subspace clusters with at least mo objects, thus, can be transformed to mining frequent itemsets from the attribute lists database L as in Table 6 with the support threshold $\eta = mo$. Here threshold mo is the parameter to control the minimum number of objects which a subspace cluster must contain. Additionally, we add another user-defined threshold ma, for conversely controlling the minimum number of attributes in a subspace cluster.

For example, if $\eta = mo = 2$ and $ma = 2$, the itemset composed of attribute symbols $\{b_0, c_1\}$ is supported by the attribute lists of objects 1 and 3 (with object 1's attribute list being $\{a_2, b_0, b_2, c_1, c_2\}$ and object 3's being $\{a_1, a_2, b_0, c_0, c_1\}$), because $\{b_0, c_1\} \subset \{a_2, b_0, b_2, c_1, c_2\}$ and $\{b_0, c_1\} \subset \{a_1, a_2, b_0, c_0, c_1\}$. Thus, we have found a subspace cluster $(T, D) = (\{1, 3\}, \{b_0, c_1\})$.

However, to find all subspace clusters according to Definition 1—an additional step beyond the original frequent itemset mining algorithm is required—and, is why we refer to it as "quasi-" frequent itemset mining. Let the set of objects which support a particular itemset α be defined as the *supporting set*. The additional step is this: once an itemset α, composed of attribute symbols, has been found to be frequent, we must create a subspace cluster for each subset of the supporting set F and α, for which the number of objects in is greater than mo. For example, if a certain itemset α is found to be frequent, and its supporting set $F = \{0, 1, 2\}$ (with $mo = 2$), then the creation of the following subspace clusters is made: $(\{0, 1\}, \alpha)$, $(\{0, 2\}, \alpha)$, $(\{1, 2\}, \alpha)$, and $(\{0, 1, 2\}, \alpha)$. Recall from Section 2, that subspace clusters exhibit downward closure; thus, all subclusters of a given subspace cluster, are themselves valid subspace clusters, and do not require any additional relative distance calculations. Using the aforementioned method, no subspace clusters are skipped.

Finally, if the quasi-frequent itemset mining algorithm were run against attribute list L with $mo = 2$ and $ma = 2$, the following subspace clusters would be discovered (with subscripts removed): $(\{1, 3\}, \{a, b\})$, $(\{1, 3\}, \{a, b, c\})$, $(\{1, 3\}, \{a, c\})$, $(\{0, 3\}, \{b, c\})$, $(\{1, 3\}, \{b, c\})$, and $(\{1, 2\}, \{b, c\})$.

In the next section, it is the goal of our algorithm to mine for these subspace clusters, identify each cluster's interval, and prune away those subspace clusters that are not relaxed and closed.

3.2 Mining

In Figure 1, we present an algorithm for the mining of relaxed closed subspace clusters.

We use C to represent the set of relaxed closed clusters. Each relaxed closed cluster in C is a triplet (z, α, I) such that z is the set of objects of the cluster, α the set of attributes of the cluster, I the interval the cluster belongs to. We also use C_i to denote i-th related relaxed closed cluster in C $(i = 1, \cdots, ||C||)$. Furthermore, the object and attribute set together, in its traditional form (T, D), of C_i is denoted as $C_i.(z, \alpha)$, and the interval of C_i is denoted by $C_i.I$.

The algorithm starts by mining the frequent itemsets from L with support support threshold $\eta = mo$, the minimum number of objects in a cluster. The frequent itemset mined is the set of attribute symbols, denoted by α, on which the objects are close with each other (with relative distance less than δ). The supporting set of α is denoted by F. All the pairs (α, F) are stored in set P. For each (α, F) from P such that α contains at least ma attributes and each subset z of F such that z contains at least mo objects, we calculate the diameter of cluster (z, α), ξ_α^z, and the interval I it belongs to. The new cluster (z, α, I) is put into cluster set C. Lastly, the newly added subspace cluster $C_{||C||}$ is compared with all previously found C_i, to see if $C_i.(z, \alpha) \subset C_{||C||}.(z, \alpha)$ and $C_i.I \equiv C_{||C||}.I$, in which case the sub-cluster C_i is pruned. If at lease one C_i is found to satisfy this test, then there is no need to check the converse, because it is an impossible occurrence. On the other hand, if no C_i was found to satisfy the previous test, then the converse is checked, that is, if

```
procedure $RCSC()$ {
  $C \leftarrow \emptyset$;
  $P \leftarrow \{(\alpha, F) |\ \alpha$ is a frequent itemset from L with
     $\eta = mo$ and $F$ its supporting set };
  foreach $(\alpha, F) \in P$ such that $||\alpha|| \geq ma$ do
      foreach $z \in \mathcal{P}(F)$ such that $||z|| \geq mo$ do
        - Calculate interval $I$ based on $\xi^z_\alpha$;
        $C \leftarrow C \cup \{(z, \alpha, I)\}$;
        $flag \leftarrow true$;
        for ($i \leftarrow 1$; $i \leq ||C|| - 1$; $i{+}{+}$) do
          if ($C_i.(z,\alpha) \subset C_{||C||}.(z,\alpha) \wedge C_i.I \equiv C_{||C||}.I$) then
            - Mark $C_i$ as pruned.;
            $flag \leftarrow false$;
          endif
        endfor
        if ($flag$) then
          for ($i \leftarrow 1$; $i \leq ||C|| - 1$; $i{+}{+}$) do
            if ($C_i.(z,\alpha) \supset C_{||C||}.(z,\alpha) \wedge C_i.I \equiv C_{||C||}.I$) then
              - Mark $C_{||C||}$ as pruned.;
              break;
            end if
          end for
        end if
      end foreach
  end foreach
}
```

Figure 1: Relaxed Closed Subspace Clustering Algorithm

$C_{||C||}.(z,\alpha) \subset C_i.(z,\alpha)$ and $C_i.I \equiv C_{||C||}.I$, and if found to be true, $C_{||C||}$ is pruned. At the end of this algorithm, C will contain only those subspace clusters that are relaxed and closed—according to Definition 4.

Finally, running the *RCSC* algorithm on our running example database S, which results in attribute list L with $mo = 2, ma = 2, \delta = 0.51, p = 3$, and $\omega = 1.0$, the following relaxed and closed subspace clusters will be found: $(\{1,3\},\{a,c\},1)$, $(\{1,3\},\{a,b,c\},2)$, $(\{0,3\},\{b,c\},2)$, and $(\{1,2\},\{b,c\},2)$.

4. EXPERIMENTAL RESULTS

In this section, we present some experimental results from an implementation of the previously described RCSC algorithm. All experiments where run on a Quad-Core Intel Xeon (R) processor of 2.0GHz with 4096 KB of cache, a total 4GB of memory, and Red Hat's Enterprise Linux Server Release 5.4 operating system. Synthetic and random datasets where created using the MSBVAR [7] package for the open-source statistical software R [20]—more specifically the function *rmultnorm* found in that package. All datasets generated follow a standard normal distribution, with no covariance between random variables / attributes. The following parameters are used in our experiments:

- T: The number of objects in the dataset, i.e. $|T|$.
- D: The number of attributes in the dataset, i.e. |D|.
- δ: The relative distance threshold.
- p: The relaxation factor (i.e., the number of divisions the interval [0-δ] should be divided into).
- ω: The overlapping threshold.
- mo: The minimum number of objects a subspace cluster should have.
- ma: The minimum number of attributes a subspace cluster should have.

Figure 2 succinctly summarizes 40 datasets, of varying T from 10 to 100 and D from 2 to 5 (with the other parameters' values seen above the figure). In that figure one can see the exponential nature of the algorithm.

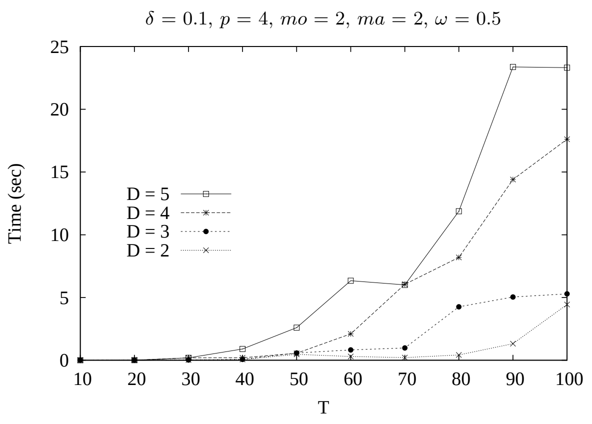

Figure 2: Synthetic and Random Datasets

Figure 3 shows the effect of p on the number of clusters found by the algorithm.

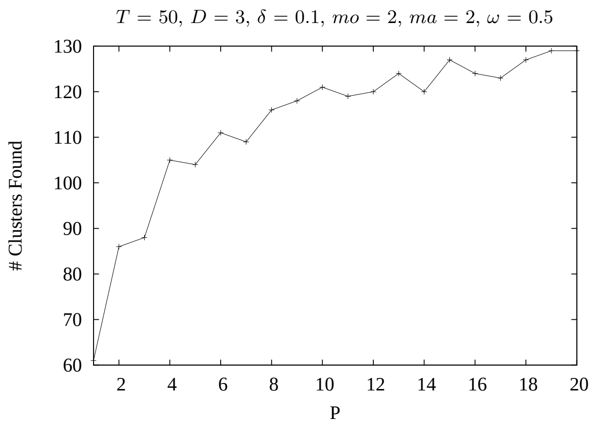

Figure 3: The Effects of P on Clusters Found

As was discussed in Section 2: as p increases, the problem reduces to mining for simple subspace clusters; we see that the number of clusters found approaches a maximum of 132—the number of clusters that would be found if performing the classic subspace clustering algorithm.

5. CONCLUSIONS AND FUTURE WORK

The main contribution of this paper, was the formulation of the new definition of relaxed closed subspace clusters. An algorithm for mining such clusters was also presented. In the experimental evaluation, which demonstrated the algorithm's time complexity and some of its properties, the algorithm proved to be very sensitive to small changes in dataset sizes and/or thresholds (particularly δ). This is most likely due to the sudden increase in the number of entries in the maximal object list M, as those parameters (and others) that could cause its increase exert their influence. It is that sensitivety, which limited the evaluation of more threshold values.

Because this is the first attempt in the creation of an algorithm to mine a newly defined problem, more efficient algorithms could no doubt be created. More efficient algorithms could only help with the apparent extreme exponential nature of the presented algorithm.

6. REFERENCES

[1] E. Achtert, C. Bohm, H.-P. Kriegel, P. Kröger, I. Müller-Gorman, and A. Zimek. Detection and visualization of subspace cluster hierarchies. In *DASFAA*, volume 4443 of *Lecture Notes in Computer Science*, pages 152–163. Springer, 2007.

[2] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, and J. S. Park. Fast algorithms for projected clustering. In *SIGMOD '99: Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 61–72, New York, NY, USA, 1999. ACM.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. *SIGMOD Rec.*, 27(2):94–105, 1998.

[4] M. R. Anderberg. *Cluster analysis for applications*. Academic Press, New York, 1973.

[5] R. E. Bellman. *Adaptive control processes: A guided tour*. Princeton University Press, Princeton, New Jersey, U.S.A., 1961.

[6] C. Bohm, K. Kailing, H.-P. Kriegel, and P. Kroger. Density connected clustering with local subspace preferences. In *ICDM '04: Proceedings of the Fourth IEEE International Conference on Data Mining*, pages 27–34, Washington, DC, USA, 2004. IEEE Computer Society.

[7] P. T. Brandt. *MSBVAR: Markov-Switching, Bayesian, Vector Autoregression Models*, 2009. R package version 0.4.0.

[8] Y. Chi, H. Wang, P. S. Yu, and R. R. Muntz. Catch the moment: maintaining closed frequent itemsets over a data stream sliding window. *Knowl. Inf. Syst.*, 10(3):265–294, 2006.

[9] C. Domeniconi, D. Papadopoulos, D. Gunopulos, and S. Ma. Subspace clustering of high dimensional data. In *Proceedings of the SIAM International Conference on Data Mining (SDM) 2004*, pages 517–521, 2004.

[10] J. H. Friedman and J. J. Meulman. Clustering objects on subsets of attributes. *Journal of the Royal Statistical Society*, 66:815–849, 2004.

[11] J. Hartigan. *Clustering Algorithms*. John Wiley and Sons, New York, 1975.

[12] N. Jardine and R. Sibson. *Mathematical Taxonomy*. Wiley London, New York,, 1971.

[13] H.-P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data*, 3(1):1–58, 2009.

[14] G. Liu, J. Li, K. Sim, and L. Wong. Distance based subspace clustering with flexible dimension partitioning. In *ICDE*, pages 1250–1254, 2007.

[15] C. Lucchese, S. Orlando, and R. Perego. Fast and memory efficient mining of frequent closed itemsets. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):21–36, 2006.

[16] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[17] I. V. Mechelen, H. H. Bock, and P. D. Boeck. Two-mode clustering methods: a structured overview. *Statistical methods in medical research*, 13(5):363–394, Oct. 2004.

[18] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.*, 6(1):90–105, 2004.

[19] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A monte carlo algorithm for fast projective clustering. In *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 418–427, New York, NY, USA, 2002. ACM.

[20] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[21] G. Song, D. Yang, B. Cui, B. Zheng, Y. Liu, and K. Xie. Claim: An efficient method for relaxed frequent closed itemsets mining over stream data. In *DASFAA*, pages 664–675, 2007.

[22] A. Tanay, R. Sharan, and R. Shamir. Biclustering algorithms: A survey. In *In Handbook of Computational Molecular Biology*, 2005.

[23] M. L. Yiu and N. Mamoulis. Iterative projected clustering by subspace mining. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):176–189, 2005.