P3C: A Robust Projected Clustering Algorithm

Gabriela Moise
Dept. of Computing Science
University of Alberta
gabi@cs.ualberta.ca

Jörg Sander
Dept. of Computing Science
University of Alberta
joerg@cs.ualberta.ca

Martin Ester
School of Computing Science
Simon Fraser University
ester@cs.sfu.ca

Abstract

Projected clustering has emerged as a possible solution to the challenges associated with clustering in high dimensional data. A projected cluster is a subset of points together with a subset of attributes, such that the cluster points project onto a small range of values in each of these attributes, and are uniformly distributed in the remaining attributes. Existing algorithms for projected clustering rely on parameters whose appropriate values are difficult to set by the user, or are unable to identify projected clusters with few relevant attributes.

In this paper, we present a robust algorithm for projected clustering that can effectively discover projected clusters in the data while minimizing the number of parameters required as input. In contrast to all previous approaches, our algorithm can discover, under very general conditions, the true number of projected clusters. We show through an extensive experimental evaluation that our algorithm: (1) significantly outperforms existing algorithms for projected clustering in terms of accuracy; (2) is effective in detecting very low-dimensional projected clusters embedded in high dimensional spaces; (3) is effective in detecting clusters with varying orientation in their relevant subspaces; (4) is scalable with respect to large data sets and high number of dimensions.

1 Introduction

Projected clustering has been mainly motivated by seminal research showing that, as the dimensionality increases, the farthest neighbor of a point is expected to be almost as close as its nearest neighbor for a wide range of data distributions and distance functions [6]. Due to this lack of contrast in distances, the concept of proximity, and subsequently the concept of a "cluster", are seriously challenged in high dimensional spaces. At the same time, irrelevant attributes are as important a motivation as the number of attributes for projected clustering. Even in data sets with moderate dimensionality, clusters may exist in *subspaces*, which are defined as subsets of attributes. The irrelevant attributes may in fact "hide" the clusters by making two objects that belong to the same cluster look as dissimilar as an arbitrary pair of objects. Furthermore, data objects may cluster differently in varying subspaces.

Traditional feature selection techniques are not effective in this scenario, because they may remove attributes that are relevant for some clusters and it may not be possible to recover those clusters in the remaining attributes [10]. Global feature transformation techniques (e.g., PCA), preserve to some extent the information from irrelevant attributes, and they may thus be unable to identify clusters that exist in different subspaces [10].

Projected clustering assumes that meaningful structure can be detected only when data is projected onto subspaces of lower dimensionality. Virtually all existing projected clustering algorithms (PROCLUS [1], DOC/FASTDOC [11], HARP [14], SSPC [15], EPCH [9]) assume, explicitly or implicitly, the following definition of a projected cluster.

Definition 1. Given a database D of d-dimensional points. A *projected cluster* is defined as a pair (X_i, Y_i), where (1) X_i is a subset of D, and (2) Y_i is a subset of attributes so that the projection of the points in X_i along each attribute $a \in Y_i$ has a small variance, compared to the variance of the whole data set on a, and (3) the points in X_i are uniformly distributed along every other attribute not in Y_i.

For a projected cluster (X_i, Y_i), the attributes in Y_i are called the "relevant" attributes for X_i, whereas the remaining attributes are called "irrelevant" attributes for X_i. The data model in projected clustering assumes that the data consists of k projected clusters, $\{(X_i, Y_i)\}_{i=\overline{1,k}}$ [1], and a set of outliers, O, where $\{X_1, \ldots, X_k, O\}$ form a partition of D. The subsets of attributes $\{Y_i\}_{i=\overline{1,k}}$ may not be disjoint and they may have different cardinalities. The outliers O are assumed to be uniformly distributed throughout the space. The *projected clustering* problem is to detect k projected clusters in the data, plus possibly a set of outliers.

[1] Notation $i = \overline{1,k}$ denotes all integers i between 1 and k.

Proceedings of the Sixth International Conference on Data Mining (ICDM'06)
0-7695-2701-9/06 $20.00 © 2006 IEEE

Definition 1 states that the relevant attributes Y_i of a projected cluster (X_i, Y_i) are a subset of the data attributes. Such projected clusters are easily interpretable by the user because the original attributes of the data set have specific meaning in real-life applications. ORCLUS [2] generalizes projected clusters (X_i, Y_i) by assuming that Y_i is an arbitrary set of orthogonal vectors.

Projected clustering is related to *subspace clustering* [3] in that both detect clusters of objects that exist in subspaces of a data set. In contrast to projected clustering, subspace clustering detects clusters of objects in all subspaces of a data set and tends to produce a large number of overlapping clusters. Related problems have been addressed in the bi-clustering community [8], where (sub)sets of objects are considered similar if they follow similar "rise-and-fall" patterns across a (sub)set of attributes.

The performance of existing projected clustering algorithms depends greatly on (1) a series of parameters whose appropriate values are difficult to anticipate by the users (e.g., the true number of projected clusters or the average dimensionality of subspaces where clusters exist), or (2) the computation of k initial clusters, which is typically performed in full dimensional space based on various heuristics. The performance of the algorithms that fall within the second category depends on how well the initial clusters approximate projected clusters in the data. These algorithms are likely to be less effective in the practically most interesting case of projected clusters with very few relevant attributes, because the members of such clusters are likely to have low similarity in full dimensional space.

In this paper, we propose an algorithm for mining projected clusters, called P3C (**P**rojected **C**lustering via **C**luster **C**ores) with the following properties.

- P3C effectively discovers the projected clusters in the data while being remarkably robust to the only parameter that it takes as input. Setting this parameter requires little prior knowledge about the data, and, in contrast to all previous approaches, there is no need to provide the number of projected clusters as input, since our algorithm can discover, under very general conditions, the true number of projected clusters.
- P3C effectively discovers very low-dimensional projected clusters embedded in high dimensional spaces.
- P3C effectively discovers clusters with varying orientation in their relevant subspaces.
- P3C is scalable with respect to large data sets and high number of dimensions.

P3C is comprised of several steps. First, regions corresponding to projections of clusters onto single attributes are computed. Second, *cluster cores* are identified by spatial areas that (1) are described by a combination of the detected regions and (2) contain an unexpectedly large number of points. Third, cluster cores are iteratively refined into projected clusters. Finally, the outliers are identified, and the relevant attributes for each cluster are determined.

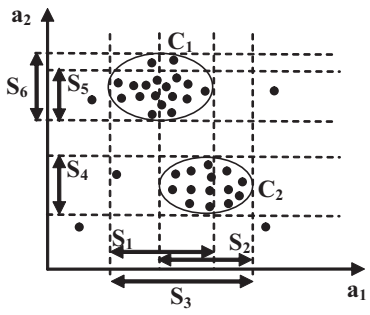

Figure 1: Overlapping true p-signatures

The remainder of the paper is organized as follows. Section 2 introduces preliminary definitions. Section 3 describes our algorithm. Section 4 presents an extensive experimental evaluation of P3C. Section 5 reviews work relevant for the this paper. Section 6 concludes the paper.

2 Preliminary Definitions

To present our algorithm for finding projected clusters, we introduce the following notation and definitions.

Let $D = (x_{ij})_{i=\overline{1,n},j=\overline{1,d}}$ be a data set of n d-dimensional data objects. Let $A = \{a_1, \ldots, a_d\}$ be the set of all attributes of the objects in D. We can assume, without restricting the generality, that all attributes have normalized values, i.e., $(x_{ij})_{i=\overline{1,n},j=\overline{1,d}} \in [0, 1]$.

An ***interval*** $S = [v_l, v_u]$ on attribute a_j is defined as all real values x on a_j so that $v_l \leq x \leq v_u$. The width of interval S is defined as $width(S) := v_u - v_l$. The attribute of an interval S is denoted by $attr(S)$, i.e., $attr(S) = a_j$, if $S \subseteq a_j$. In figure 1, S_1, S_2, and S_3 are intervals on attribute a_1, S_4, S_5 and S_6 are intervals on attribute a_2, $attr(S_1) = attr(S_2) = attr(S_3) = a_1$, and $attr(S_4) = attr(S_5) = attr(S_6) = a_2$. To ease the presentation, we specify the attribute of an interval only when it is necessary.

Let S be an interval on attribute a_j. The *support set* of S, denoted by $SuppSet(S)$, represents the set of database objects that belong to S, i.e., $SuppSet(S) := \{x \in D | x.a_j \in S\}$. The *support* of S, denoted by $Supp(S)$, is the cardinality of its support set, i.e., $Supp(S) := |SuppSet(S)|$.

A ***p-signature*** $\mathbf{S}$ is defined as a set $\mathbf{S} = \{S_1, \ldots, S_p\}$ of p intervals on some (sub)set of p *distinct* attributes $\{a_{j_1}, \ldots, a_{j_p}\}$ ($j_i \in \{1, \ldots, d\}$), where $attr(S_i) = a_{j_i}$. S_i is also called the *projection* of $\mathbf{S}$ onto attribute a_{j_i}, $i = \overline{1, p}$. For example, in figure 1, $\mathbf{S} = \{S_3, S_4\}$ is a 2-signature, where S_3 is the projection of $\mathbf{S}$ onto attribute a_1, and S_4 is the projection of $\mathbf{S}$ onto attribute a_2. $\{S_3, S_1\}$ is not a 2-signature, because S_3 and S_1 are intervals on the same attribute a_1.

Proceedings of the Sixth International Conference on Data Mining (ICDM'06)
0-7695-2701-9/06 $20.00 © 2006 IEEE

The *support set* of a p-signature $\mathbf{S} = \{S_1, \ldots, S_p\}$, denoted by $SuppSet(\mathbf{S})$, represents the set of database objects that are contained in the support sets of all intervals in $\mathbf{S}$, i.e., $SuppSet(\mathbf{S}) := \{x \in D | x \in \bigcap_{i=1}^{p} SuppSet(S_i)\}$. The *support* of a p-signature $\mathbf{S}$, denoted by $Supp(\mathbf{S})$, is the cardinality of its support set, i.e., $Supp(\mathbf{S}) := |SuppSet(\mathbf{S})|$.

A ***true p-signature*** $\tilde{\mathbf{S}}$ of a projected cluster (X_i, Y_i), $Y_i = \{a_1, \ldots, a_p\}$, is a p-signature $\{S_1, \ldots, S_p\}$, where S_i is the smallest interval on attribute a_i that contains the projections onto a_i of all the points in X_i, $i = \overline{1, p}$. Figure 1 illustrates two projected clusters, C_1, and C_2, both having a_1 and a_2 as the only relevant attributes. The true p-signature of C_1 is the 2-signature $\{S_1, S_6\}$, and the true p-signature of C_2 is the 2-signature $\{S_2, S_4\}$.

Since an attribute may be relevant to more than one projected cluster, true p-signatures may *overlap*, i.e., they may contain overlapping intervals. In figure 1, C_1 and C_2 have overlapping true p-signatures, since intervals S_1 and S_2 overlap on attribute a_1. We assume that true p-signatures can overlap as long as they are not *nested* within each other. True p-signatures $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{R}}$ are *nested* if for every interval S_i in $\tilde{\mathbf{S}}$, there is an interval S_j in $\tilde{\mathbf{R}}$ so that $S_i \subseteq S_j$.

3 Algorithm P3C

P3C is based on the idea that if the true p-signatures of projected clusters were known, then clusters can be immediately computed as the support sets of the true p-signatures. Since the true p-signatures are not known, P3C computes in two steps a set of p-signatures that match or approximate well the true p-signatures of projected clusters in the data. First, on every attribute, intervals that match or approximate well projections of true p-signatures onto that attribute are computed (section 3.1). Second, the challenge is to determine which intervals actually represent the same true p-signature. P3C addresses this challenge by aggregating the computed intervals into *cluster cores*. Roughly speaking, a cluster core consists of a p-signature $\mathbf{S}$ and its support set $SuppSet(\mathbf{S})$, so that the p-signature $\mathbf{S}$ approximates a true p-signature $\tilde{\mathbf{S}}$ of a projected cluster C, and a large fraction of the points in $SuppSet(\mathbf{S})$ belongs to C (section 3.2).

For the example in figure 1, P3C first computes the interval S_3 on attribute a_1 that approximates the projections of the true 2-signatures $\{S_1, S_6\}$ and $\{S_2, S_4\}$ onto attribute a_1, and intervals S_5 and S_4 that approximate/match the projections of the same true 2-signatures onto attribute a_2. Second, P3C aggregates these intervals into two cluster cores, i.e., $\{S_3, S_4\}$ and $\{S_3, S_5\}$, which can be regarded as approximations of the two projected clusters in the data.

Cluster cores may include in their support sets additional points that do not belong to the projected clusters that they approximate. This happens when the intervals are wider than the projections of true p-signatures that they approximate. In figure 1, interval S_3 is wider than interval S_2, and thus, the support set of cluster core $\{S_3, S_4\}$ includes points that do not belong to cluster C_2. On the other hand, cluster cores may not include completely in their support sets the projected clusters that they approximate. This is the case when the intervals are tighter than the projections of true p-signatures that they approximate. In figure 1, interval S_5 is tighter than interval S_6, and thus the support set of cluster core $\{S_3, S_5\}$ does not include all points of cluster C_1. Thus, in order to compute the projected clusters, the supports sets of cluster cores are iteratively refined (section 3.3). Finally, outliers are detected (section 3.4), and relevant attributes for each cluster are determined (section 3.5).

3.1 Projections of true p-signatures

This section describes how P3C computes, for each attribute, intervals that match or approximate well projections of true p-signatures onto that attribute.

An attribute that is irrelevant for all projected clusters exhibits, by definition 1, uniform distribution. In contrast, an attribute that is relevant for at least one projected cluster will exhibit in general a non-uniform distribution, because it contains one or more intervals with unusual high support corresponding to projections of clusters onto that attribute.

Note that theoretically an attribute could exhibit uniform distribution, even though it is relevant for several projected clusters. This is the case when projected clusters are constructed in such a way that their projections on a specific attribute have equal support, and thus they form a uniform histogram. In such cases, it may still be possible to recover p-signatures of the involved clusters, which are incomplete, but can be later refined - unless projected clusters are constructed in such a way that all their relevant attributes look uniform. However, it is assumed that these situations are not common in typical applications for projected clustering.

We need to identify attributes with uniform distribution, and, for the non-uniform attributes, to identify intervals with unusual high support. For this task, the *Chi-square* goodness-of-fit test [13] is employed. Each attribute is divided into the same number of equi-sized *bins*. Sturge's rule [13] suggests that the number of bins should be equal to $\lfloor 1 + log_2(n) \rfloor$, where n is the number of data objects. For every bin in every attribute, its support is computed. The Chi-square test statistic sums, over all bins in an attribute, the squared difference between the bin support and the average bin support, normalized by the average bin support. Based on the Chi-square statistic, the uniform attributes are determined at a confidence level of $\alpha = 0.001$. The confidence level α does not act as a parameter of our method. α is set to one of the standard values used in statistical hypothesis testing: the value 0.001 signifies that the probability of declaring an attribute non-uniform when in fact the attribute

is uniform is very small, i.e., less than 0.001.

On the attributes deemed non-uniform, the bin with the largest support is *marked*. The remaining un-marked bins are tested again using the Chi-square test for uniform distribution. If the Chi-square test indicates that the un-marked bins "look" uniform, then we stop. Otherwise, the bin with the second-largest support is marked. Then, we repeat testing the remaining un-marked bins for the uniform distribution and marking bins in decreasing order of support, until the current set of un-marked bins satisfies the Chi-square test for uniform distribution. At this point, intervals are computed by merging adjacent marked bins. The process of marking bins is linear in the number of bins.

The computed intervals may be wider or tighter than the projections of true p-signatures that they approximate. Overlapping true p-signatures may lead to the former case (e.g., intervals S_1 and S_2 are approximated by interval S_3). An example of the latter case is an interval that approximates the projection of a true p-signature onto an attribute where the cluster is normally distributed. In this case, the interval may only capture the most dense region of the projection (e.g., interval S_5 on attribute a_2).

3.2 Cluster cores

In figure 1, the computed intervals form only two possible 2-signatures, $\{S_3, S_5\}$ and $\{S_3, S_4\}$, which actually represent the two projected clusters C_1 and C_2. However, in practical applications, the number of possible p-signatures that can be constructed from the set of computed intervals is large. The challenge is to determine which p-signatures do in fact represent projected clusters. This section describes how P3C addresses this challenge.

Let **S** be a p-signature. Let **R** = **S** $\cup \{S'\}$ be a $(p+1)$-signature composed of **S** and an interval S' that is not in **S**. Assuming that **S** is a subset of some true t-signature **T** ($t > p$), we could ask the question whether S' also belongs to **T**. When S' does belong to **T**, the support $Supp(\mathbf{R})$ of **R** is likely to have a larger value than in the case when S' does *not* belong to **T**, because, in the former case, $Supp(\mathbf{R})$ should include a large fraction of the projected cluster with signature **T**. Clearly, the support $Supp(\mathbf{R})$ of **R** = **S** $\cup \{S'\}$ is equal to the number of points in $SuppSet(\mathbf{S})$ that also belong to S'. Therefore, we want to compute how many points in $SuppSet(\mathbf{S})$ are expected to belong to S' in the case when S' does *not* belong to **T**.

The points in $SuppSet(\mathbf{S})$ are mainly points of a projected cluster with signature **T**, and interval S' does not belong to **T**. In this case, under the assumption that points in $SuppSet(\mathbf{S})$ are uniformly distributed in the attribute of interval S', the expected number of points in $SuppSet(\mathbf{S})$ that also belong to S' is proportional to $width(S')$. The following definition formally introduces the notion of *expected support* of a $(p+1)$-signature **R** = **S** $\cup \{S'\}$ with respect to a p-signature **S** obtained by adding interval S' to **S**.

Definition 2. Let **S** be a p-signature. Let **R** = **S** $\cup \{S'\}$ be a $(p+1)$-signature composed of **S** and interval S' (S' not in **S**). The ***expected support*** of the $(p+1)$-signature **R** given the p-signature **S**, denoted by ESupp(**R** = **S** $\cup \{S'\}|$**S**), is defined as:

$$\text{ESupp}(\mathbf{R} = \mathbf{S} \cup \{S'\}|\mathbf{S}) := \text{Supp}(\mathbf{S}) * \text{width}(S')$$

We consider that if the actual support $Supp(\mathbf{R})$ of **R** is *significantly larger* than the expected support ESupp(**R** = **S** $\cup \{S'\}|$**S**) of **R** given **S**, then this is evidence that S' belongs to the same true t-signature as **S**.

We need a quantitative way of deciding when the observed support $Supp(\mathbf{R})$ of **R** = **S** $\cup \{S'\}$ is *significantly larger* than the expected support ESupp(**R** = **S** $\cup \{S'\}|$**S**) of **R** given **S**. For this task, we employ the Poisson probability density function $Poisson(v, E)$ of observing v occurrences of a certain event within a time interval/spatial region, given the expected number E of random occurrences per time interval/spatial region [13]:

$$Poisson(v, E) := \frac{exp(-E) * E^v}{v!}$$

where *exp* stands for the exponential function. In our case, we measure the probability of observing a certain number of points (i.e., $Supp(\mathbf{R})$) within a spatial region, given the expected number of points within this spatial region (i.e., ESupp(**R** = **S** $\cup \{S'\}|$**S**)), under a random process that uniformly distributes the points in $SuppSet(\mathbf{S})$ onto the attribute of S'.

We call an observed support *significantly larger* than an expected support, if the observed support is larger than the expected support, and the Poisson probability of observing the support given the expected support is smaller than a certain value, which we call the *Poisson threshold*.

The Poisson probability quantifies how likely the observed support Supp(**R**) of **R** is with respect to the expected support ESupp(**R** = **S** $\cup \{S'\}|$**S**) of **R** given **S**: the less likely the observed support, the stronger the evidence that S' represents the same projected cluster as **S**. The Poisson threshold is the only "parameter" required by P3C. The Poisson threshold is different from typical parameters used by clustering algorithms (such as the number of clusters) in that it requires little prior knowledge about the data. The Poisson threshold signifies the error probability that the user is willing to accept. Concretely, the value $1.0E-20$ for the Poisson threshold signifies that the probability of declaring that S' represents the same projected cluster as **S**, when in fact this is not true, is very small, i.e., less than $1.0E-20$. This is why higher values for the Poisson threshold like $1.0E-1$ are not useful. On the other hand, a very small value for the Poisson threshold would result in failing to recognize that S' represents the same projected cluster as **S**, when in fact

this is true. The robustness of P3C to the Poisson threshold is studied empirically in section 4 (see figure 2).

Intuitively, a p-signature $\mathbf{S} = \{S_1, \ldots, S_p\}$ represents a projected cluster C if $\mathbf{S}$ consists of (1) *only* and (2) *all* intervals that represent cluster C. The first condition is equivalent to requesting that for any q-signature $\mathbf{Q} \subseteq \mathbf{S}$ ($q = \overline{1, p-1}$), and any interval $S' \in \mathbf{S} \setminus \mathbf{Q}$, there is evidence that S' represents the same projected cluster as $\mathbf{Q}$. The second condition is equivalent to requesting that $\mathbf{S}$ is *maximal*, i.e., for any interval S' not in $\mathbf{S}$, there is no evidence that S' represents the same projected cluster as $\mathbf{S}$. Formally, a cluster core can be defined as following.

Definition 3. A p-signature $\mathbf{S} = \{S_1, \ldots, S_p\}$ together with its support set $SuppSet(\mathbf{S})$ is called a ***cluster core***, if:

1. For any q-signature $\mathbf{Q} \subseteq \mathbf{S}$, $q = \overline{1, p-1}$, and any interval $S' \in \mathbf{S} \setminus \mathbf{Q}$, it holds that:
   Supp($\mathbf{Q} \cup \{S'\}$) $>$ ESupp ($\mathbf{Q} \cup \{S'\}|\mathbf{Q}$), and
   Poisson(Supp($\mathbf{Q} \cup \{S'\}$), ESupp($\mathbf{Q} \cup \{S'\}|\mathbf{Q}$)) $<$ Poisson _threshold
2. For any interval S' not in $\mathbf{S}$, it holds that
   Supp($\mathbf{S} \cup \{S'\}$) $\leq$ ESupp ($\mathbf{S} \cup \{S'\}|\mathbf{S}$), or
   Poisson(Supp($\mathbf{S} \cup \{S'\}$), ESupp($\mathbf{S} \cup \{S'\}|\mathbf{S}$)) $\geq$ Poisson_threshold

Condition 1 in definition 3 is equivalent to requesting, for any q-signature $\mathbf{Q} \subseteq \mathbf{S}$ ($q = \overline{1, p-1}$), and any interval $S' \in \mathbf{S} \setminus \mathbf{Q}$, that Supp($\mathbf{Q} \cup \{S'\}$) is significantly larger than ESupp ($\mathbf{Q} \cup \{S'\}|\mathbf{Q}$). Condition 2 in definition 3 is equivalent to requesting, for any interval S' not in $\mathbf{S}$, that Supp($\mathbf{S} \cup \{S'\}$) is *not* significantly larger than ESupp ($\mathbf{S} \cup \{S'\}|\mathbf{S}$).

Condition 1 in definition 3 is anti-monotonic, in the sense that, given a p-signature $\mathbf{S}$ that satisfies condition 1, any sub-signature of $\mathbf{S}$ also satisfies condition 1. This fact motivates an Apriori-like generation of p-signatures that satisfy condition 1. Condition 1 acts like the support test in frequent itemsets generation [4]: a signature consisting of $(q+1)$ intervals will not be generated if any of its sub-signatures consisting of q intervals does not satisfy condition 1. p-signatures that satisfy condition 1 are generated, and the ones that are "maximal" in the sense of condition 2 are reported as cluster cores.

3.3 Computing projected clusters

Let k be the number of cluster cores constructed according to section 3.2. The support sets of these cluster cores may not necessarily contain all and only the points of the projected clusters that the cluster cores approximate, depending on the accuracy of the intervals computed in section 3.1. In this section, we discuss how P3C refines the k cluster cores into k projected clusters.

The refinement of k cluster cores into k projected clusters is performed in a subspace of (reduced) dimensionality d' of the original d-dimensional data, containing all attributes that were deemed non-uniform according to the analysis presented in section 3.1.

The support sets of the k cluster cores are not necessarily disjoint, because they may contain, in addition to the members of the clusters approximated by the cluster cores, outlier objects and/or other clusters' members that have the signatures of other cluster cores. The membership of data points to cluster cores can be described through a fuzzy membership matrix $M = (m_{il})_{i=\overline{1,n}, l=\overline{1,k}}$, where m_{il} denotes the membership of object i to cluster core l; it is defined as follows: $m_{il} = 0$, if data point i does not belong to the support set of any cluster core; m_{il} is equal to the fraction of clusters cores that contain data point i in their support set, if i is in the support set of cluster core l.

We want to compute for each data point its probability of belonging to each projected cluster using the Expectation Maximization (EM) algorithm [7]. For this purpose, we will initialize EM with the fuzzy membership matrix M. Since the fuzzy membership matrix M contains unassigned data points, i.e., data points with membership 0 everywhere, we first assign these points to one of the k cluster cores.

In the case of projected clusters, by definition 1, cluster members project closely to cluster means on the directions with the least spread. Thus, cluster members have shorter Mahalanobis distances to cluster means than non-cluster members. Provided that the support set of a cluster core mainly consists of members of the projected cluster C approximated by the cluster core, data points with short Mahalanobis distance to the mean of the support set are highly likely to be members of C. Based on these considerations, unassigned data points are assigned to the "closest" cluster core in terms of Mahalanobis distances to means of support sets of cluster cores.

Once all unassigned points have been assigned to cluster cores, the fuzzy membership matrix M is equivalent to a fuzzy partition of the data points into k projected clusters. EM computes data points' probabilities of belonging to projected clusters based on Mahalanobis distances between data points and the means of projected clusters. Therefore, cluster members have higher probabilities of belonging to their clusters than non-cluster members. EM is considered to converge when the means of the projected clusters remain unchanged between two consecutive iterations. Typically, when starting with cluster cores, it takes only 5 to 10 iterations until convergence, since the cluster cores typically approximate well projected clusters in the data.

The output of EM is a matrix of probabilities that gives for each data point its probability of belonging to each projected cluster. Since the data model in projected clustering assumes disjoint projected clusters, we convert the matrix of probabilities produced by EM into a hard membership matrix by assigning each data point to the most probable pro-

jected cluster. Interestingly, our method can also be used to discover *overlapping* clusters. In this respect, P3C positions itself between projected and subspace clustering.

3.4 Outlier Detection

Although each data point has been assigned to a projected cluster in section 3.3, the data set may contain outlier points that need to be identified. We use a standard technique for multivariate outlier detection [12]. The Mahalanobis distances between data points and the means of the projected clusters to which they belong are compared to the critical value of the Chi-square distribution with d' degrees of freedom at a confidence level of $\alpha = 0.001$. The confidence level α signifies that the probability of failing to recognize a true outlier is less than 0.001. Data points with Mahalanobis distances larger than this critical value are declared outliers.

3.5 Relevant Attributes Detection

Once the cluster members have been identified, the relevant attributes for each projected cluster can be determined. The relevant attributes of a projected cluster include the attributes of the intervals that make up the p-signature of the cluster core based on which this cluster has been computed. As discussed in section 3.1, an attribute may be considered uniform although it may be relevant for several projected clusters. To cover these rather rare cases too, we test, for each projected cluster, using the Chi-square test, whether its members are uniformly distributed in the attributes initially deemed uniform. When the members of a projected cluster are not uniformly distributed in one of the attributes initially considered uniform, then those attributes are included in the attributes considered relevant for the projected cluster. Finally, the p-signatures of projected clusters can be refined by computing for each relevant attribute the smallest interval that the cluster members project onto.

4 Experimental Evaluation

The experiments reported in this section were conducted on a Linux machine with 3 GHz CPU and 2 GB RAM.

Synthetic Data. Synthetic data sets were generated as described in [1], [2], with $n = 10,000$ data points, $d = 100$ attributes, 5 clusters with sizes 15% to 25% of n, and 5% of n outliers. The performance of P3C is studied based on the following criteria:

1. Distribution of cluster points in the relevant subspace: *uniform* versus *normal*.
2. Projected clusters having an *equal* number of relevant attributes versus projected clusters having *different* numbers of relevant attributes.

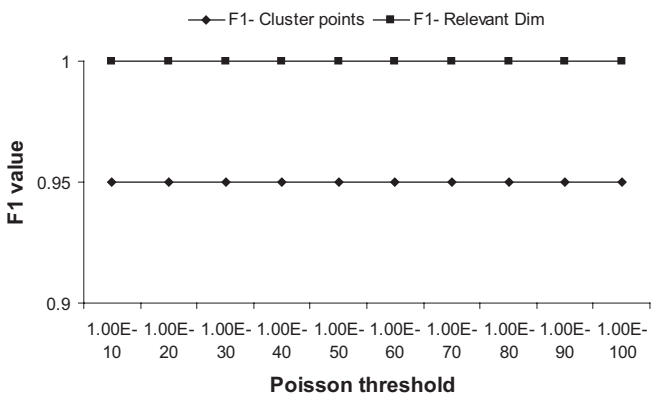

Figure 2: P3C's sensitivity to the Poisson threshold

3. Projected clusters with *axis-parallel* orientation versus projected clusters with *arbitrary* orientation.

Combining these three criteria results in 8 categories of data sets. A data set in the category "Uniform_Equal_Parallel" is a data set for which the cluster points are *uniformly* distributed in the relevant subspace, the number of relevant attributes for each projected cluster is *equal*, and the eigenvectors of each projected cluster's covariance matrix are *parallel* to the coordinate axes. In each category, we generated data sets with average cluster dimensionality 2%, 4%, 6%, 8%, 10%, 15%, and 20% of the data dimensionality d. In total, 56 synthetic data sets have been generated.

For data sets where cluster points are normally distributed in their relevant subspace, we ensured that the variance of cluster members on individual relevant attributes is between 1% and 10% of the variance of all data points when uniformly distributed on an attribute. Various amounts of overlap were introduced among the signatures of projected clusters, i.e., the larger the average cluster dimensionality, the higher the chance for the overlap between signatures.

Real Data. We have tested P3C on two real-world data sets. The first data set is the colon cancer data set of Alon et al. [5] that measures the expression level of 40 tumor and 22 normal colon tissue samples in 2000 human genes. The task is to discover projected clusters using samples as data objects and genes as attributes. This task is challenging due to the data sparsity (i.e., 62 data points in 2000 attributes), but of practical importance. A relevant attribute of a projected cluster represents a gene that has similar values in the samples that belong to the projected cluster. Provided that a projected cluster contains mainly tumor or normal samples, the relevant attributes are potential indicators for the presence, respectively absence, of colon cancer.

Projected clusters may exist in data sets with moderate dimensionality when some of the attribute are irrelevant. The second data set is the Boston housing data [2], which consists of 12 numerical attributes of 506 suburbs of Boston.

[2] http://www.ics.uci.edu/ mlearn/MLRepository.html

Since this data set is not labeled, we apply clustering in an exploratory fashion, and report interesting findings.

Experimental setup. We evaluate the performance of P3C against the following competing algorithms for projected clustering [3]: PROCLUS [1], FASTOC [11], HARP [14], SSPC [15], and ORCLUS [2].

P3C requires only one parameter setting, namely the Poisson threshold. P3C does *not* require the user to set the target number of clusters; instead, it discovers a certain number of clusters by itself. In contrast, all competing algorithms require the user to specify the target number of clusters.

On synthetic data, we have run the competing algorithms with the target number of clusters equal to the true number of projected clusters. PROCLUS and ORCLUS require the average cluster dimensionality as a parameter, which was set to the true average cluster dimensionality. HARP requires the maximum percentage of outliers as a parameter, which was set to the true percentage of outliers. For FASTDOC and SSPC, several reasonable values for their parameters were tried, and we report results for the parameter settings that consistently produced the best accuracy. SSPC was run without any semi-supervision. Except HARP, all competing algorithms are non-deterministic; thus each of them is run five times, and the results are averaged.

On the colon cancer data, we have run the competing algorithms with the target number of clusters equal to the number of classes (i.e., 2). Multiple values were tried for the other parameters required by the competing algorithms, and the results with the best accuracy are reported. Since this data set contains no points labeled as outliers, the outlier removal option of all algorithms was disabled.

On the housing data, since it has no labels, the evaluation of the competing algorithms is cumbersome. The reason is that the performance of the competing algorithms is dependent on a large number of required parameters, including critical ones such as the number of clusters and the average cluster dimensionality. Under these circumstances, we apply only P3C on the second real data set.

Performance measures. We refer to true clusters as *input* clusters, and to found clusters as *output* clusters. On synthetic data, cluster labels and relevant attributes for each cluster are known. On the colon cancer data, only the cluster labels are known. We use an $F1$_value to measure the clustering accuracy. For each output cluster i, we determine the input cluster j^i with which it shares the largest number of data points. The *precision* of output cluster i is defined as the number of data points common to i and j^i divided by the total number of data points in i. The *recall* of output cluster i is defined as the number of data points common to i and j^i divided by the total number of data points in j^i. The $F1$_value of output cluster i is the harmonic mean of its precision and recall. The $F1$_value of a clustering solution is obtained by averaging the $F1$_values of all its output clusters. Similarly, we use an $F1$_value to measure the accuracy of found relevant attributes based on the matching between output and input clusters.

Sensitivity analysis. We have studied the sensitivity of P3C to the Poisson threshold. Figure 2 illustrates the accuracy of P3C measured using the two $F1$_values introduced above for one of our synthetic data sets as the Poisson threshold is progressively decreased from $1.0E-10$ to $1.0E-100$. We observe that P3C is remarkably robust with respect to the Poisson threshold. Similar results have been obtained on all our synthetic data sets, but are omitted due to space limitations. Consequently, we have set the Poisson threshold at $1.0E-20$.

Accuracy results. On synthetic data, in all the performed experiments, the number of clusters discovered by P3C equals the true number of projected clusters in the data.

Figures 3 to 10 show the accuracies of the compared algorithms as a function of increasing average cluster dimensionality for the 8 categories of data sets. We observe that P3C significantly and consistently outperforms the competing projected clustering algorithms, both in terms of clustering accuracy and in terms of accuracy of the found relevant attributes.

The difference in performance between P3C and previous methods is particularly large for data sets that contain very low-dimensional projected clusters embedded in high dimensional spaces. Even in these difficult cases P3C shows very high accuracies, in contrast to the modest accuracies obtained by the competing algorithms. As the average cluster dimensionality increases, the accuracy of the competing algorithms increases as well.

Our experiments indicate that P3C effectively discovers projected clusters with varying orientation in their relevant subspaces. The accuracy of P3C on data sets where projected clusters have axis-parallel orientation is as high as the accuracy of P3C on data sets where projected clusters have arbitrary orientation.

The accuracy of P3C on data sets where projected clusters are uniformly distributed in their relevant subspaces is slightly higher than the accuracy of P3C on data sets where projected clusters are normally distributed in their relevant subspaces. The reason is that projections of clusters onto their relevant attributes can be approximated more faithfully by the computed intervals for clusters in the former category than for clusters in the latter category.

The number of relevant attributes for projected clusters does not have an impact on the performance of P3C. This is to be expected, since P3C does not use in any way the

[3] We intended to compare with EPCH [9] too, but after consulting with its authors, and using the original implementation, we could not find a parameter setting that produces results with reasonable accuracy on our synthetic data sets.

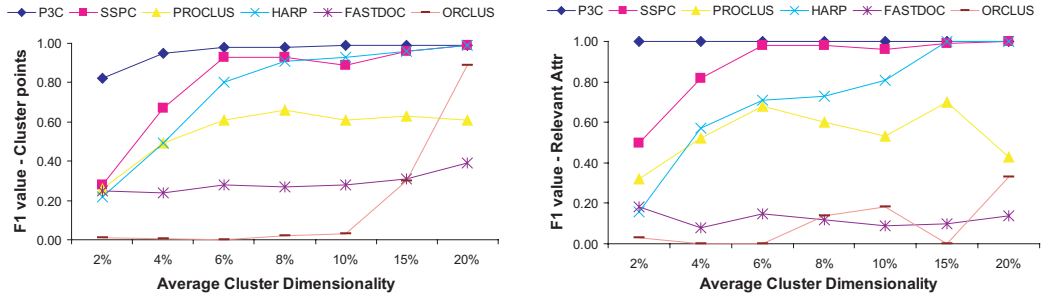

Figure 3: Category Uniform_Equal_Parallel

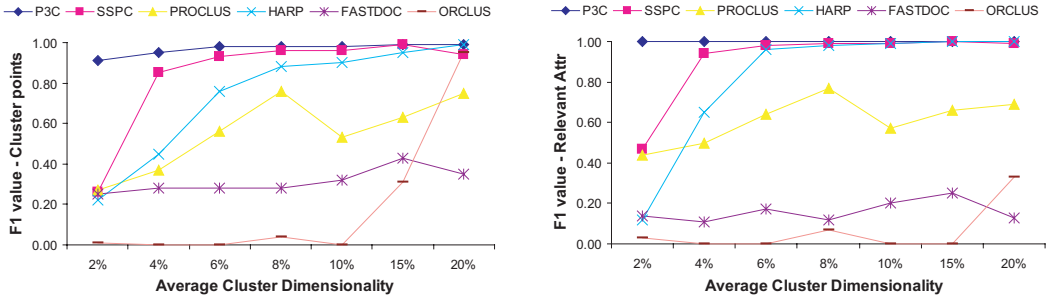

Figure 4: Category Uniform_Equal_NonParallel

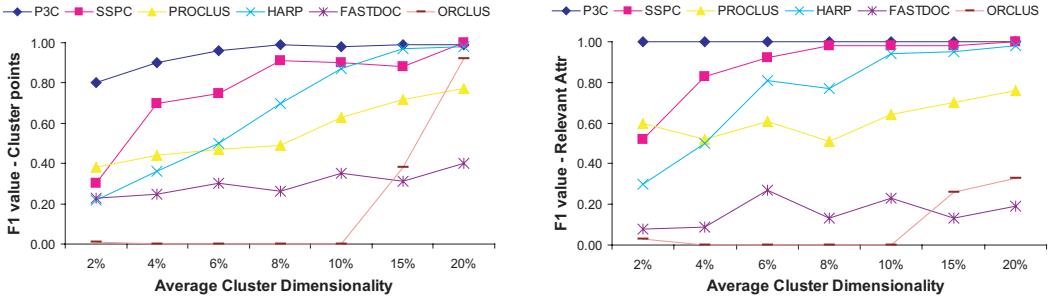

Figure 5: Category Normal_Equal_Parallel

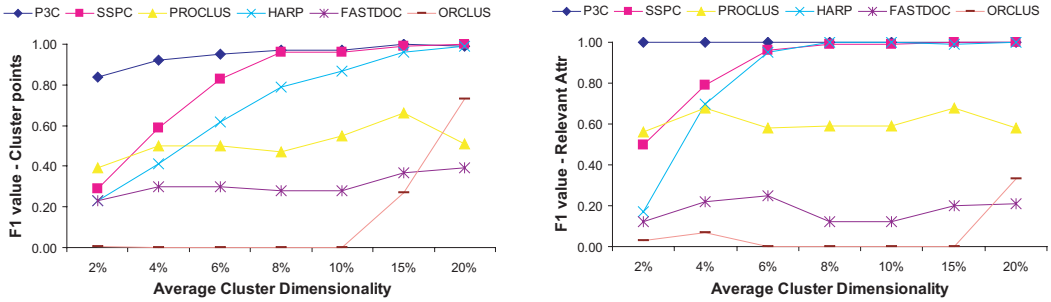

Figure 6: Category Normal_Equal_NonParallel

Proceedings of the Sixth International Conference on Data Mining (ICDM'06)
0-7695-2701-9/06 $20.00 © 2006 IEEE

average cluster dimensionality. Interestingly, the accuracy of the found relevant attributes is 100% in all experiments.

On the colon cancer data, P3C discovers 2 projected clusters. P3C obtains the highest clustering accuracy (67%), followed by HARP (55%) and SSPC (53%), whereas the accuracies of the other projected clustering algorithms are significantly lower on this data set: FASTDOC and PROCLUS obtain 43% accuracy, and ORCLUS obtains 35%. The dimensionality of these 2 projected clusters in 11, which is much smaller than the dimensionality of the data set (i.e., 2000). This indicates that only a relatively small fraction of genes out of the total number of genes may be relevant for distinguishing between cancer and normal tissues, as also noted in previous work [5]. The biological significance of the genes selected as relevant is yet to be determined.

On the housing data, P3C discovers 2 projected clusters, which exist in subspaces of dimensionality 4. The first projected cluster contains suburbs that are similar in terms of residential land, crime rate, pollution and property tax. The second projected cluster contains suburbs that are similar in terms of business land, size, distance to employment centers, and property tax. This data set illustrates that projected clusters can exist in data sets with a moderate number of attributes when some of these attributes are irrelevant. To verify that the members of the 2 projected clusters are not close in full dimensional space, we have run KMeans ($k = 2$) several times. In all runs, the members of the projected clusters discovered by P3C are distributed between the clusters found by KMeans, which indicate that full dimensional clustering cannot reproduce the same clusters.

Robustness to outliers. Data sets with $n = 10,000$, $d = 100$, 5 clusters, average cluster dimensionality 4, and different percentages of outliers were generated. Figure 11 shows the accuracies of the compared algorithms as a function of increasing percentages of outliers. P3C, as well as the competing algorithms, are robust in the presence of outliers. The clustering accuracy of P3C decreases only slightly as more outliers are introduced. Even when the percentage of outliers in the data is as high as 25%, P3C still obtains a clustering accuracy of 86%. The accuracy of the found relevant attributes of P3C remains 100% with increasing percentages of outliers.

Scalability experiments. In all scalability figures, the time is represented on a log scale.

Figure 12 shows scalability results for data sets with $d = 10$, 2 clusters, 5% outliers, average cluster dimensionality 2, and increasing database sizes. The scalability of P3C with respect to database size is comparable to the scalability of the fastest projected clustering algorithms.

Figure 13 shows scalability results for data sets with $n = 10,000$, 2 clusters, 5% outliers, average cluster dimensionality 2, and increasing database dimensionalities. P3C is relatively unaffected by increasing data dimensionality, because attributes with uniform distributions are not involved in the computation of cluster cores.

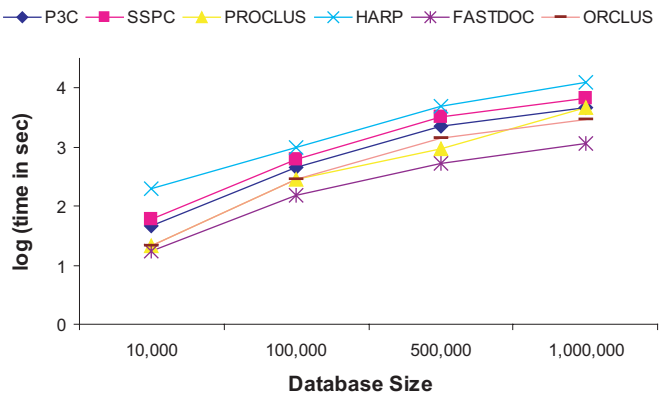

Figure 12: Scalability with increasing database size

Figure 14 shows scalability results for data sets with $n = 10,000$, $d = 100$, 5 clusters, 5% outliers, and increasing average cluster dimensionalities. The running time of P3C increases with increasing average cluster dimensionality, due to the increased complexity of p-signatures generation. However, as the average cluster dimensionality increases, clusters become increasingly detectable in full dimensional space. P3C has comparable running times to the other projected clustering algorithms at low average cluster dimensionality, which is the critical cases that "full-dimensional" clustering algorithms cannot deal with.

In summary, P3C consistently and significantly outperforms existing projected clustering algorithms in terms of clustering accuracy and accuracy of the found relevant attributes, while being as efficient as the fastest of these algorithms on data sets with low-dimensional projected clusters.

5 Related Work

PROCLUS [1] is essentially a k-medoid algorithm adapted to projected clustering. A main difference to the standard k-medoid algorithm is that initial clusters around the medoids have to be computed as basis for the simultaneous selection of relevant attributes. The performance of PROCLUS crucially depends on 2 required input parameters (k - the desired number of projected clusters, and l - the average cluster dimensionality), whose appropriate values are difficult to guess. Another weakness is the strong dependency on the initial clustering which is hard to determine since it is performed in full-dimensional space where the "true" distances will be distorted by noisy attributes.

ORCLUS [2] is a generalization of PROCLUS that can discover clusters in arbitrary sets of orthogonal vectors. The quality of a projected cluster is defined as the sum of the variances of the cluster members along the projected attributes. Therefore, in order to identify the projection in

Proceedings of the Sixth International Conference on Data Mining (ICDM'06)
0-7695-2701-9/06 $20.00 © 2006 IEEE

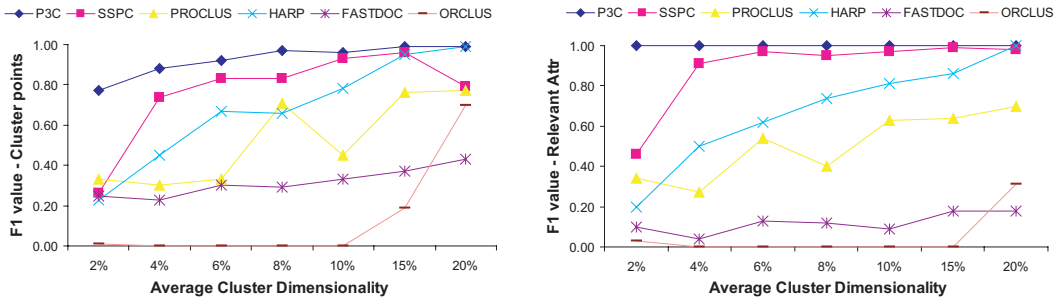

Figure 7: Category Uniform_Different_Parallel

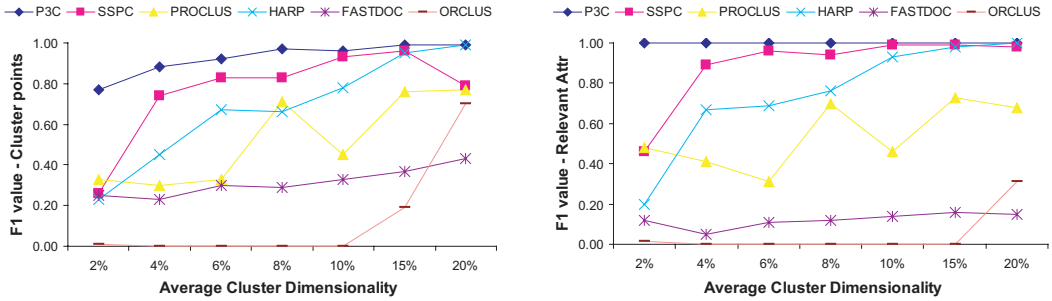

Figure 8: Category Uniform_Different_NonParallel

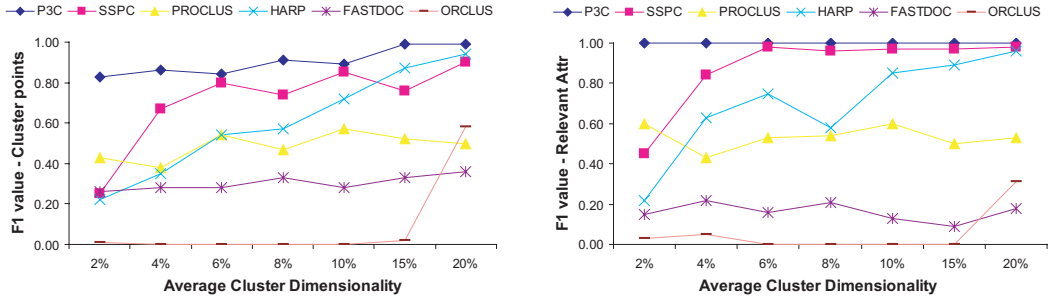

Figure 9: Category Normal_Different_Parallel

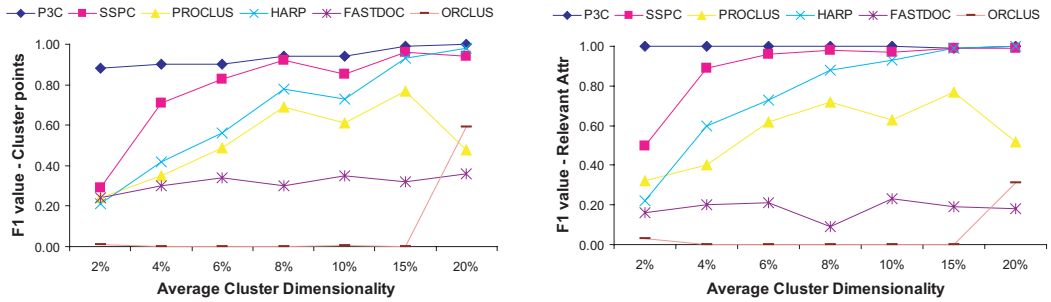

Figure 10: Category Normal_Different_NonParallel

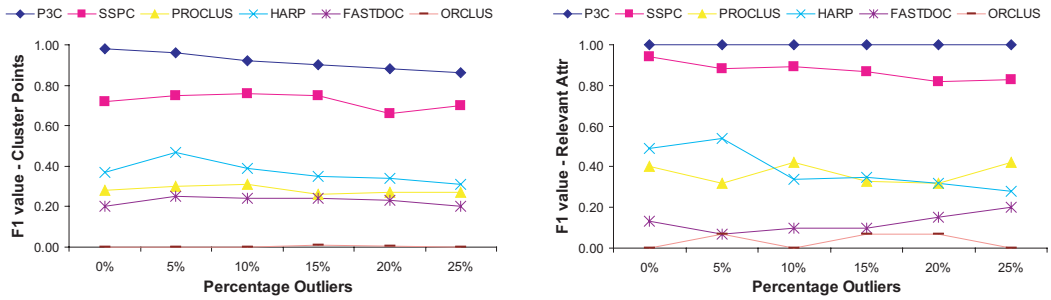

Figure 11: Robustness to Noise

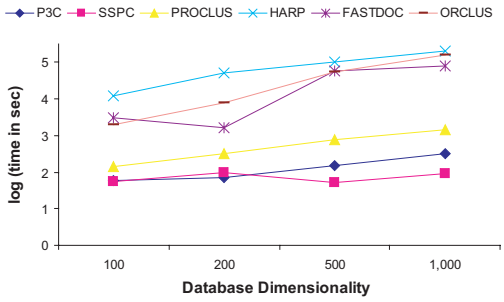

Figure 13: Scalability with increasing database dim

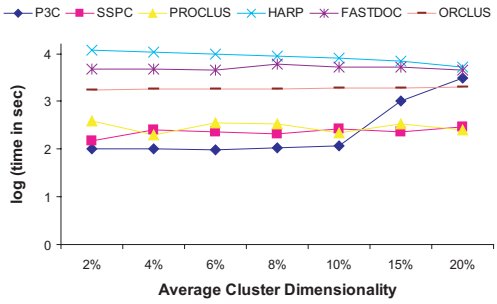

Figure 14: Scalability with increasing avg. cluster dim

which a set of points cluster "best" according to the quality measure, ORCLUS selects the eigenvectors corresponding to the smallest eigenvalues of the covariance matrix of the given set of points. The parameter l is used to decide how many such eigenvectors to select. While ORCLUS can find significantly more general clusters, it inherits the weaknesses of PROCLUS discussed above.

DOC [11] defines a projected cluster as a pair (C, D), where C is a subset of points, and D is a subset of attributes, such that C contains at least a fraction α of the total number of points, and D consists of all the attributes on which the projection of C is contained within a segment of length w. DOC defines the function μ to measure the quality of a projected cluster as $\mu(|C|, |D|) = |C| * (1/\beta)^{|D|}$ where β is a user-specified parameter that controls the trade-off between the number of data points and the number of relevant attributes in a projected cluster. DOC computes one projected cluster at a time, optimizing its quality using a randomized algorithm with certain quality guarantees. In order to reduce the time complexity of DOC, its authors introduce a variant, called FASTDOC, which uses three heuristics to reduce the search time. Similar to PROCLUS, the performance of DOC is sensitive to the choice of the input parameters, whose values are difficult to determine for real-life data sets. In addition, the assumption that a projected cluster is a hyper-cube of same side length in all attributes may not be appropriate in real applications.

HARP [14] is an agglomerative, hierarchical clustering algorithm that starts by placing each data object in a cluster. Two clusters are allowed to merge if the resulting cluster has d_{min} or more relevant attributes, and an attribute is selected as relevant for the merged cluster if a given relevance score is greater than R_{min}. d_{min} and R_{min} are two internal thresholds that start at some harsh values so that only objects belonging to the same real cluster are likely to be merged. Subsequently, as the clusters increase in size, and the relevant attributes are more reliably determined, the two thresholds are progressively decreased, until they reach some base values or a certain number of clusters has been obtained. HARP avoids some of the problems of the previous approaches, such as the computation of initial clusters, or the usage of parameters whose values are difficult to set. However, HARP inherits the drawbacks of hierarchical clustering algorithms, in particular the lack of backtracking in the clustering process and the quadratic runtime complexity which makes it not scalable to large data sets.

Yip et al. proposes the algorithm*SSPC* [15], similar in structure to PROCLUS, and whose performance can be improved by the use of domain knowledge in the form of labeled objects and/or labeled attributes. The algorithm uses an objective function based on the relevance score of HARP [14]. The quality of a clustering solution is the sum of the qualities of each individual cluster, and the quality of an individual cluster is the sum of the relevance scores of the cluster's relevant attributes. The performance of *SSPC* depends on a user-defined parameter that controls

Proceedings of the Sixth International Conference on Data Mining (ICDM'06)
0-7695-2701-9/06 $20.00 © 2006 IEEE

the relevance scores of attributes. *SSPC* can find projected clusters with moderately low dimensionality whereas most other methods fail due to an initialization based on the full-dimensional space.

EPCH [9] computes low-dimensional histograms ($1D$ or $2D$), and "dense" regions are identified in each histogram, based on iteratively lowering a threshold that depends on a user-specified parameter. For each data object, a "signature" is derived, which consists of the identifiers of the dense regions the data object belongs to. The similarity between two objects is measured by the matching coefficient of their signatures in which zero entries in both signatures are ignored. Objects are grouped in decreasing order of similarity until at most a user-specified number of clusters is obtained. *EPCH* differs from our method both in how the computation of low-dimensional projections of projected clusters is performed, and in how these projections are used to recover projected clusters. In particular, dense regions from different attributes are not combined into higher-dimensional regions, but used to measure the similarity of pairs of objects. In addition, the performance of *EPCH* is sensitive to the values of its parameters.

6 Conclusions

Projected clustering is motivated by data sets with a large number of attributes or with irrelevant attributes. Existing projected clustering algorithms crucially depend on user parameters whose appropriate values are often difficult to anticipate, and are unable to discover low-dimensional projected clusters. In this paper, we address these drawbacks through the novel, robust projected clustering algorithm P3C. P3C is based on the computation of so-called cluster cores. Cluster cores are defined as regions of the data space containing an unexpectedly high number of points, forming cores of actual projected clusters. Cluster cores are generated in an Apriori-like fashion, and subsequently refined into projected clusters. Lastly, outliers are removed and the relevant cluster attributes are detected. Our experimental evaluation on numerous synthetic data sets and two real data sets demonstrates that P3C can indeed discover projected clusters, including clusters in very low-dimensional subspaces, and clusters with varying orientation, distribution or number of relevant attributes, while being robust to the only required parameter. P3C consistently outperforms the state-of-the-art methods in terms of accuracy, and it is robust to noise. In addition, our algorithm scales well with respect to large data sets and high number of dimensions.

As future work, we will investigate the extension of P3C for categorical data.

Acknowledgments. We would like to thank Kevin Yip from Yale University for providing us with the implementation of the comparing algorithms for projected clustering. This research was supported by the Alberta Ingenuity Fund and the iCORE Circle of Research Excellence.

References

[1] C. C. Aggarwal, C. Procopiuc, J. L. Wolf, P. S. Yu, and J. S. Park. Fast algorithms for projected clustering. In *SIGMOD*, 1999.

[2] C. C. Aggarwal and P. S. Yu. Finding generalized projected clusters in high dimensional spaces. In *SIGMOD*, 2000.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD*, 1998.

[4] R. Agrawal and R. Srikan. Fast Algorithms for Mining Association Rules. In *VLDB*, 1994.

[5] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays. *PNAS*, 96:6745–6750, 1999.

[6] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is nearest neighbor meaningful? *LNCS*, 1540:217–235, 1999.

[7] A. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood for incomplete data via the EM algorithm. *J. R. Stat. Soc.*, 39:1–38, 1977.

[8] S. C. Madeira and A. J. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE TCBB*, 1(1):24–45, 2004.

[9] E. Ng, A. Fu, and R. Wong. Projective clustering by histograms. *IEEE TKDE*, 17(3):369–383, 2005.

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

[11] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A Monte Carlo algorithm for fast projective clustering. In *SIGMOD*, 2002.

[12] P. J. Rousseeuw and B. C. V. Zomeren. Unmasking multivariate outliers and leverage points. *J. Amer. Stat. Assoc.*, 85(411):633–651, 1990.

[13] G. W. Snedecor and W. G. Cochran. *Statistical Methods*. Iowa State University Press, 1989.

[14] K. Y. Yip, D. W. Cheung, and M. K. Ng. HARP: A practical projected clustering algorithm. *IEEE TKDE*, 16(11):1387–1397, 2004.

[15] K. Y. Yip, D. W. Cheung, and M. K. Ng. On discovery of extremely low-dimensional clusters using semi-supervised projected clustering. In *ICDE*, 2005.