Knowl Inf Syst
DOI 10.1007/s10115-007-0090-6

REGULAR PAPER

Robust projected clustering

Gabriela Moise · Jörg Sander · Martin Ester

Received: 18 December 2006 / Revised: 23 March 2007 / Accepted: 23 April 2007

Abstract Projected clustering partitions a data set into several disjoint clusters, plus outliers, so that each cluster exists in a subspace. Subspace clustering enumerates clusters of objects in all subspaces of a data set, and it tends to produce many overlapping clusters. Such algorithms have been extensively studied for numerical data, but only a few have been proposed for categorical data. Typical drawbacks of existing projected and subspace clustering algorithms for numerical or categorical data are that they rely on parameters whose appropriate values are difficult to set appropriately or that they are unable to identify projected clusters with few relevant attributes. We present P3C, a robust algorithm for projected clustering that can effectively discover projected clusters in the data while minimizing the number of required parameters. P3C does not need the number of projected clusters as input, and can discover, under very general conditions, the true number of projected clusters. P3C is effective in detecting very low-dimensional projected clusters embedded in high dimensional spaces. P3C positions itself between projected and subspace clustering in that it can compute both disjoint or overlapping clusters. P3C is the first projected clustering algorithm for both numerical and categorical data.

Keywords Projected clustering · Subspace clustering · Clustering numerical and categorical data

1 Introduction

Traditionally, clustering algorithms measure the similarity between data objects by considering all features/attributes of a data set. These approaches are successful for low-dimensional data sets [11]. However, in high-dimensional data sets, traditional clustering algorithms tend

G. Moise (✉) · J. Sander
Department of Computing Science, University of Alberta, Edmonton, AB T6G 2E8, Canada
e-mail: gabi@cs.ualberta.ca

M. Ester
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada

to break down both in terms of accuracy, as well as efficiency, due to the so-called "curse of dimensionality". Seminal research [7] has shown 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. 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, not all attributes are relevant for cluster analysis. 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.

These facts motivated the idea that, in high dimensional spaces, data objects may form clusters only when a subset of the attributes is considered. A subset of attributes is referred to as a *subspace*. Furthermore, data objects may cluster differently in varying subspaces.

For instance, consider a gene expression data set that measures the expression level of human genes in several human tissues. When clustering tissues, we deal with a clustering problem in a very high dimensional space, because the number of genes, typically in the thousands, is several orders of magnitude larger than the number of tissues, usually in the tens. Because of the sparse nature of the data, it is unlikely that data points, representing tissues, form clusters in full dimensional space. Instead, data points may form clusters only when a small number of "relevant" attributes are considered. As noted in the bio-medical literature, only a relatively small number of genes out of the total number of genes may be relevant for distinguishing between normal and cancerous tissues. Furthermore, data points may form different clusters in different subsets of attributes, depending on the different phenotypes represented by these attributes. For example, the cancerous tissues may form a cluster when a certain subset of attributes is selected, whereas the normal tissues may form a cluster when a different subset of attributes is selected. The selected attributes are potential indicators for the presence, respectively absence, of cancer.

Global dimensionality reduction techniques such as feature selection and feature transformation (e.g., PCA) are not effective in this scenario. These techniques cluster data only in a particular subspace, in which it may not be possible to recover all clusters [17]. Besides, information concerning objects clustered differently in different subspaces is lost.

Projected clustering is defined to deal with the aforementioned challenges. Virtually all existing projected clustering algorithms assume, explicitly or implicitly, the following definition of a projected cluster.

Definition 1.1 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 points in X_i project along each attribute $a \in Y_i$ onto a small range of values, compared to the range of values 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.

Definition 1.1 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.

Existing projected clustering algorithms are either based on the computation of k initial clusters in full dimensional space, or leverage the idea that clusters with as many relevant attributes as possible are preferable. Consequently, 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. Furthermore, a re-occurring weakness of these algorithms is that their performance depends greatly on 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). Finally, projected clusters are, by definition, disjoint. However, a data point may satisfy the signature of more than one projected cluster. Projected clustering is not able to capture this type of information.

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 according to their respective definition of a cluster. A large number of overlapping clusters is typically reported. Existing subspace clustering algorithms start with $1D$ clusters, which are subsequently merged bottom-up, similarly to the Apriori algorithm for finding frequent itemsets [4], in order to compute clusters of higher dimensionality. To avoid an exhaustive search through all possible subspaces, most subspace clustering algorithms use a global density threshold that ensures the downward closure property necessary for the application of an Apriori style search. However, using a global density threshold ignores the fact that the sparseness of the data increases with dimensionality.

The majority of existing subspace and projected clustering algorithms are designed for *numerical* data sets, i.e., data sets where the domain of every attribute is inherently ordered. However, many real data sets are *categorical*, i.e., the attribute domains are discrete and not ordered [22].

Subspace and projected clustering algorithms designed for numerical data are not readily applicable to categorical data sets. Some of these algorithms (e.g., PROCLUS [1]) use distance functions that exploit the geometric properties of the data space, which cannot be effectively captured by categorical distance functions, such as the simple matching coefficient. Other algorithms (e.g., ORCLUS [2]) require numerical computations that are not well-defined for categorical attributes (e.g., mean, variance, eigenvalues). Finally, some algorithms (e.g., CLIQUE [3]) are based on the discretization of individual data attributes into bins, and the notion of "neighboring" bins is used to manage the search through all possible subspaces. Categorical attributes lack order, and thus this notion is not directly applicable.

A significantly smaller body of work has been dedicated to the subspace clustering problem on categorical data than to the same problem on numerical data. Existing subspace and projected clustering algorithms for categorical data (e.g., SUBCAD [9], CLICKS [26], CACTUS [10]) exhibit the same weaknesses as their numerical counterparts.

Contributions and outline of the paper. 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.

1. 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.
2. P3C effectively discovers very low-dimensional projected clusters embedded in high dimensional spaces.
3. Although essentially a *projected clustering* algorithm, P3C may assign a data point to more than one cluster if the point satisfies the description of more than one projected cluster.
4. P3C is the first projected clustering algorithm that can be applied on both numerical and categorical data sets.
5. P3C effectively discovers clusters with varying orientation in their relevant subspaces in the case of numerical data.
6. P3C is scalable with respect to large data sets and high number of attributes.

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 refined into projected clusters, outliers are identified, and the relevant attributes for each cluster are determined.

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 this paper. Section 6 concludes the paper.

2 Preliminary 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.

For numerical data sets, we can assume, without restricting the generality, that all attributes have normalized values, i.e., $(x_{ij})_{i=\overline{1,n}, j=\overline{1,d}} \in [0, 1]$. For categorical data sets, we assume that each attribute a_i is associated with a finite, discrete domain $dom(a_i), \forall i = \overline{1, d}$.

For a *numerical* attribute a_j, 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 $width(S) := v_u - v_l$.

For a *categorical* attribute a_j, an ***interval*** S on attribute a_j is defined as a subset of attribute values for attribute a_j, i.e., $S = \{x_{i_1 j}, \ldots, x_{i_h j}\} \subseteq dom(a_j)$. In this case, the width of the interval S is defined as the number of attribute values in S, i.e., $width(S) := h$.

The attribute of an interval S is denoted by $attr(S)$, i.e., $attr(S) = a_j$, if $S \subseteq a_j$. Figure 1 illustrates a numerical data set with two projected clusters, C_1, and C_2, both having a_1 and a_2 as the only relevant attributes. 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 simplify 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., $Supp(S) := |SuppSet(S)|$.

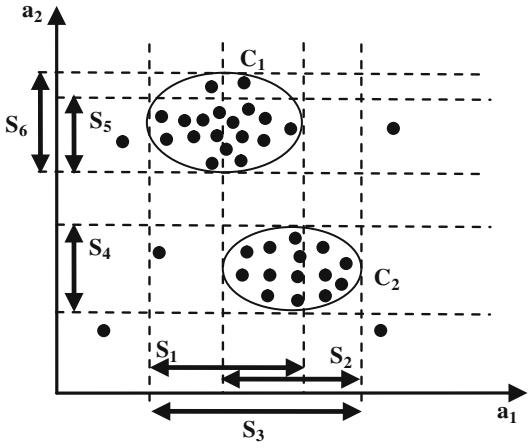

Fig. 1 Overlapping true p-signatures

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 Fig. 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.

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, $i = \overline{1, p}$. In Fig. 1, 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 Fig. 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 completely *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 (Sect. 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

Input: Data set $D = (x_{ij})_{i=\overline{1,n}, j=\overline{1,d}}$, the Poisson threshold
Output: k disjoint or overlapping projected clusters, their relevant attributes, and outliers
Method:

1. Discretize each attribute into bins (section 3.1).
2. Determine attributes with non-uniform distribution, and compute intervals that approximate projections of clusters onto these attributes (sections 3.1.1 and 3.1.2).
3. Aggregate the intervals into cluster cores (section 3.2).
4. Refine cluster cores into projected clusters, compute outliers, and detect clusters' relevant attributes (section 3.3).

Fig. 2 Pseudo-code of P3C

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 (Sect. 3.2).

For the example in Fig. 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 Fig. 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 Fig. 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 refined, outliers are detected, and relevant attributes for each cluster are determined (Sect. 3.3).

The pseudo-code of P3C is given in Fig. 2.

3.1 Approximating true p-signatures

An attribute that is irrelevant for all projected clusters exhibits, by Definition 1.1, 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.

Consequently, 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 [20] is employed. For this test, each attribute must be first discretized into *bins*.

Discretization of numerical attributes. Each numerical attribute is divided into the same number of equi-sized bins. Sturge's rule [20] suggests that the number of bins should be equal to $\lfloor 1 + log_2(n) \rfloor$, where n is the number of data objects.

Discretization of categorical attributes. Since the domain of a categorical attribute is finite, we have a bin for each single value in the domain of a categorical attribute.

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. The process of marking bins is linear in the number of bins.

The challenge is to determine, for each attribute, which marked bins correspond to the projection of the same true p-signature for which this attribute is relevant, since clusters may have overlapping projections on some relevant attributes.

3.1.1 Intervals on numerical attributes

On numerical attributes, intervals are simply computed as maximal sets of consecutive marked bins. Since the support of each marked bin deviates significantly from its expected support, the support of intervals formed with marked bins will deviate from their expected support significantly (i.e., with the average of the deviations of the constituent marked bins).

In order to understand the computation of intervals on categorical attributes in the next section, we use the natural order exhibited by numerical attributes to define an "adjacency" relationship between marked bins on the *same* attribute.

Definition 3.1 Let mb_1 and mb_2 be two marked bins on the *same* numerical attribute a_i. mb_1 and mb_2 are called "adjacent" if they are consecutive bins on attribute a_i.

Subsequently, each numerical attribute can be represented as a graph, in which the vertices are the marked bins on that attribute, and edges exist between two vertices if the corresponding marked bins are adjacent, as defined in 3.1.

Property 3.1 Let a_i be a numerical attribute. Let $mb_{i_1}, \ldots, mb_{i_h}$ be h marked bins on attribute a_i. Then, $mb_{i_1}, \ldots, mb_{i_h}$ are consecutive bins on attribute a_i if and only if $mb_{i_1}, \ldots, mb_{i_h}$ form a connected component in the associated graph.

Proof Let us assume that $mb_{i_1}, \ldots, mb_{i_h}$ are consecutive bins on attribute a_i. Then, by definition 3.1, there is an edge between mb_{i_l} and $mb_{i_{l+1}}$ in the associated graph,

$\forall l = \overline{1, h-1}$. Therefore, there is a path between any two vertices mb_{i_l} and mb_{i_j}, $\forall l, j = \overline{1, h}$, i.e., $mb_{i_1}, \ldots, mb_{i_h}$ form a connected component. Conversely, let as assume that $mb_{i_1}, \ldots, mb_{i_h}$ form a connected component in the associated graph. By definition 3.1, each node $mb_{i_l} \forall l = \overline{1, h}$ must have degree at most 2, and there must exist 2 nodes with degree exactly 1. Starting from one of the nodes of degree 1, we can infer, based on definition 3.1, the order of the bins $mb_{i_1}, \ldots, mb_{i_h}$ on attribute a_i. Thus, $mb_{i_1}, \ldots, mb_{i_h}$ are consecutive bins.

Consequence. On numerical attributes, intervals are equivalent to connected components in the associated graph representation.

3.1.2 Intervals on categorical attributes

Categorical attributes lack order, and thus the notion of "consecutive" bins is not applicable. In order to determine, for each attribute, which marked bins should be merged within the same interval, we introduce a criterion based on the Poisson distribution.

The Poisson-based criterion. Let $\mathbf{S}$ be a p-signature. Let $\mathbf{R} = \mathbf{S} \cup \{S'\}$ be a $(p+1)$-signature composed of $\mathbf{S}$ and an interval S' that is not in $\mathbf{S}$. Assuming that $\mathbf{S}$ is a subset of some true t-signature $\mathbf{T}$ ($t > p$), we could ask the question whether S' also belongs to $\mathbf{T}$.

When S' does belong to $\mathbf{T}$, the support $Supp(\mathbf{R})$ of $\mathbf{R}$ is likely to have a larger value than in the case when S' does *not* belong to $\mathbf{T}$, because, in the former case, $Supp(\mathbf{R})$ should include a large fraction of the projected cluster with signature $\mathbf{T}$. Clearly, the support $Supp(\mathbf{R})$ of $\mathbf{R} = \mathbf{S} \cup \{S'\}$ 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 $\mathbf{T}$.

The points in $SuppSet(\mathbf{S})$ are mainly points of a projected cluster with signature $\mathbf{T}$, and interval S' does not belong to $\mathbf{T}$. 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 $\mathbf{R} = \mathbf{S} \cup \{S'\}$ with respect to a p-signature $\mathbf{S}$ obtained by adding interval S' to $\mathbf{S}$.

Definition 3.2 Let $\mathbf{S}$ be a p-signature. Let $\mathbf{R} = \mathbf{S} \cup \{S'\}$ be a $(p+1)$-signature composed of $\mathbf{S}$ and interval S' (S' not in $\mathbf{S}$). The ***expected support*** of the $(p+1)$-signature $\mathbf{R}$ given the p-signature $\mathbf{S}$, denoted by ESupp($\mathbf{R} = \mathbf{S} \cup \{S'\}|\mathbf{S}$), is defined as:

$$ESupp(\mathbf{R} = \mathbf{S} \cup \{S'\}|\mathbf{S}) := Supp(\mathbf{S}) * width(S') \tag{1}$$

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

We need a quantitative way of deciding when the observed support $Supp(\mathbf{R})$ of $\mathbf{R} = \mathbf{S} \cup \{S'\}$ is *significantly larger* than the expected support ESupp($\mathbf{R} = \mathbf{S} \cup \{S'\}|\mathbf{S}$) of $\mathbf{R}$ given $\mathbf{S}$. 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 [20]:

$$Poisson(v, E) := \frac{exp(-E) * E^v}{v!} \tag{2}$$

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., $\text{ESupp}(\mathbf{R} = \mathbf{S} \cup \{S'\}|\mathbf{S})$), 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 $\text{Supp}(\mathbf{R})$ of $\mathbf{R}$ is with respect to the expected support $\text{ESupp}(\mathbf{R} = \mathbf{S} \cup \{S'\}|\mathbf{S})$ of $\mathbf{R}$ given $\mathbf{S}$: the less likely the observed support, the stronger the evidence that S' represents the same projected cluster as $\mathbf{S}$.

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 $\mathbf{S}$, 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 $\mathbf{S}$, when in fact this is true.

We use the Poisson-based criterion to decide whether two marked bins on two *distinct* categorical attributes belong to the same cluster projection onto these attributes.

Definition 3.3 Let mb_1 and mb_2 be two marked bins on two *distinct* categorical attributes a_i and a_j $(i \neq j)$, respectively. Let P be a user-specified Poisson threshold. mb_1 and mb_2 belong to the projection of the same true p-signature onto a_i and a_j if the following two conditions are satisfied:

1. $Supp(\{mb_1\} \cup \{mb_2\}) > ESupp(\{mb_1\} \cup \{mb_2\}|\{mb_1\})$, and
   $Poisson(Supp(\{mb_1\} \cup \{mb_2\}), ESupp(\{mb_1\} \cup \{mb_2\}|\{mb_1\})) < P$
2. $Supp(\{mb_2\} \cup \{mb_1\}) > ESupp(\{mb_2\} \cup \{mb_1\}|\{mb_2\})$, and
   $Poisson(Supp(\{mb_2\} \cup \{mb_1\}), ESupp(\{mb_2\} \cup \{mb_1\}|\{mb_2\})) < P$

Using Definition 3.3, all pairs of marked bins on distinct categorical attributes that represent the same cluster projections onto these attributes are computed. Subsequently, these pairs are used to define an "adjacency" relationship between marked bins on the *same* categorical attribute.

Definition 3.4 Let mb_1 and mb_2 be two marked bins on the *same* categorical attribute a_i. mb_1 and mb_2 are called "adjacent" if there is at least one marked bin mb_3 on another categorical attribute a_j, $j \neq i$, so that (mb_1, mb_3) and (mb_2, mb_3) belong to the same cluster projection according to Definition 3.3.

In analogy to numerical data, intervals on categorical attributes are defined as connected components under the adjacency relationship introduced in Definition 3.4. Some subspace clustering algorithms for categorical data (e.g., CACTUS, CLICKS) define intervals on categorical attributes as cliques under a different adjacency criterion. Note that cliques under the adjacency criterion in Definition 3.4 will produce overlapping intervals, which cannot be obtained on numerical data, according to the adjacency criterion in Definition 3.1. In order to have a unified treatment of numerical and categorical data, P3C computes connected components instead of cliques under its adjacency criterion.

3.2 Cluster cores computation

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. We use the Poisson-based criterion to address this challenge.

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 3.5 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 3.5 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 3.5 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 3.5 satisfies the downward closure property, 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 Projected clusters computation, outlier detection, and relevant attributes identification

Let k be the number of cluster cores constructed according to Sect. 3.2. The support sets of the k cluster cores are not necessarily disjoint, because data points may satisfy the signature of several cluster cores. In addition, some data points may not satisfy the signature of any cluster core. Note that the projected clusters computation and outlier detection are 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 Sect. 3.1.

Numerical data. Cluster cores correspond to axis-parallel hyper-rectangles, but, on numerical data, clusters may have non-axis parallel orientations in their relevant subspace. In such cases, even if the computed intervals match the projections of true p-signatures onto their relevant attributes, the support set of a cluster core may not necessarily contain all and only the points of the projected cluster approximated by the cluster core.

We take advantage of a property of numerical projected clusters that follows from Definition 1.1, namely that 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, we assign data points that do not satisfy the signature of any cluster core to the "closest" cluster core in terms of Mahalanobis distances to means of support sets of cluster cores. At this point, the k cluster cores correspond to a fuzzy partitioning of the data set into k clusters, which is used to initialize the Expectation Maximization (EM) algorithm [8]. 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 stops when the means of the projected clusters remain unchanged between two consecutive iterations. When starting with cluster cores, it takes only 5–10 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. If disjoint projected clusters are desired, each data point is assigned to the most probable cluster. If overlapping projected clusters are needed, a data point is assigned to a projected cluster if the probability of belonging to it is larger than $1/k$.

Clustering and outlier detection are closely related [21]. We use a standard technique for outlier detection [19]. 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 to cluster means larger than this critical value are declared outliers.

Categorical data. In this case, clusters are axis-parallel hyper-rectangles. Similarly to the computation of projected clusters on numerical data, we could use some distance function for categorical data to assign the remaining data points to cluster cores. We could then use the resulting fuzzy partitioning of the data points into k clusters to initialize EM. Based on the current definition of clusters, EM computes cluster means and covariance matrices. Such computations are not possible for categorical data. Instead, we compute the projected clusters by measuring how "relevant" the bins that appear in the signatures of cluster cores are with respect to the cluster cores.

We consider the space given by the union of all ('attribute_id', 'bin_id') pairs, where 'attribute_id' is an attribute of an interval I that appears in the definition of a cluster core, and 'bin_id' is a bin that appears in the interval I onto 'attribute_id'.

Each cluster core CC is represented as a vector in this space, and each entry in this vector represents the "relevance score" of a certain ('attribute_id', 'bin_id') pair with respect to the cluster core CC. Similarly to the standard "TF-IDF" scheme used in information retrieval, the relevance score of a pair ('attribute_id', 'bin_id') with respect to a cluster core CC consists of the product of two terms. The first term of the product is the fraction of cluster core points that belongs to the support set of 'bin_id' onto attribute 'attribute_id'. The second term of the product is the inverse of the fraction of data points that belongs to the support set of 'bin_id' onto attribute 'attribute_id'. In other words, the relevance score of a pair

('attribute_id', 'bin_id') with respect to a cluster core CC is proportional to the frequency at which the pair ('attribute_id', 'bin_id') appears among the points of the cluster core CC, and inverse proportional to the frequency at which the pair ('attribute_id', 'bin_id') appears among all data points.

Each data point is also represented as a vector in the aforementioned space. In this case, the entry corresponding to a ('attribute_id', 'bin_id') pair is either 1 or 0, depending on whether the data point belongs to the support set of 'bin_id' onto attribute 'attribute_id' or not.

The similarity between a data point and a cluster core is defined as the dot product of their corresponding vectors. We can regard the resulting similarity matrix between data points and cluster cores as a matrix of membership probabilities by normalizing each matrix entry by the sum of its row. Similarly to the numerical data, we can compute now disjoint or overlapping clusters. Outliers are the points with similarity 0 to all cluster cores.

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 Sect. 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 test indicates a non-uniform distribution, then those attributes are included in the attributes considered relevant for the projected cluster.

3.4 Discussion

P3C can compute both disjoint and overlapping clusters. In this respect, P3C positions itself between projected and subspace clustering algorithms. P3C shares some similarities with the subspace clustering algorithms in that it approximates $1D$ projections of clusters, which are subsequently aggregated in a bottom-up fashion to find the clusters. Definition 3.5 used by P3C to aggregate $1D$ cluster projections satisfies the downward closure property in order to avoid an exhaustive traversal of the search space. However, unlike the other subspace clustering algorithms, our criterion leverages the data model, and it is not a global density threshold that ignores the fact that density decreases with increasing dimensionality.

P3C can deal with data sets with numerical attributes only, or with data sets with categorical attributes only. In the case of mixed data sets, the computation of intervals can be performed according to the attribute type, as discussed in Sects. 3.1.1 and 3.1.2, followed by the cluster cores computation, as in Sect. 3.2. However, the projected clusters computation and outlier detection proposed for numerical data are not applicable on the categorical attributes, since they require numerical computations that are not well defined for categorical attributes. In this case, we regard the numerical attributes as being discretized either as suggested in Sect. 3.1, or based on existing domain knowledge, and we apply the projected clusters computation and outlier detection for categorical data.

If desired, the p-signatures of projected clusters can be refined. For numerical data, we can compute for each relevant attribute of a cluster the smallest interval that the cluster members project onto. For categorical data, we can remove from the signature of a cluster the bins with small relevance score.

4 Experimental evaluation

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

In the following, we use "P3C_hard" to refer to the variant of P3C that computes disjoint clusters, and "P3C_soft" to refer to the variant of P3C that computes overlapping clusters. We use the term P3C when we refer collectively to both variants.

Our goal is to assess the performance of P3C on data sets that contain low-dimensional projected clusters embedded in high dimensional spaces. As noted in previous work (e.g., [23]), full-dimensional clustering algorithms are likely to be less effective on these data sets, because the members of such clusters are likely to have low similarity in full dimensional space.

Synthetic Data. Synthetic data sets were generated as described in Aggarwal et al. [1], Aggarwal and Yu [2], [26] with $n = 10,000$ data points, $d = 100$ attributes, 5 clusters with sizes 15–25% of n, and 5% of n outliers. For the categorical synthetic data sets, we used the domain size = 100 (i.e., the number of categories on each attribute).

In order to study the performance of P3C on numerical data, eight categories of data sets have been generated, according to 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.
3. Projected clusters with *axis-parallel* orientation versus projected clusters with *arbitrary* orientation.

In addition, two categories of categorical data sets were generated: category "Equal" and category "Different" according to criterion (2).

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 numerical and 14 categorical 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 three real-world data sets: two numerical, and one mixed data set.

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 2,000 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 2,000 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 (UCI Machine Learning Repository[2]), which consists of 12 numerical attributes of 506 suburbs of Boston. Since this data set is not labeled, we apply clustering in an exploratory fashion, and report interesting findings.

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

To experiment on categorical data, we have chosen the Hepatitis data set (UCI Machine Learning Repository) with 155 data points classified into two classes ("die" — 32 points, and "live" — 123 points). It contains 19 attributes, out of which 5 are numerical. The numerical attributes are discretized based on medical criteria, as suggested in the documentation that comes with this data.

Experimental setup. We evaluate the performance of P3C against the following competing algorithms for projected clustering: PROCLUS [1], FASTDOC [18], MINECLUS [25], HARP [23], SSPC [24], and ORCLUS [2].[3] We intended to compare with EPCH [16] 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.

We also evaluate the performance of P3C against the subspace clustering algorithm MAFIA [15], which was shown to outperform CLIQUE [3]. We have considered both SUBCLU [12] and FIRES [13] in our preliminary experiments. However, the original implementations of these algorithms proved to have unacceptable scalability with respect to the number of objects and attributes, in the case of SUBCLU, and a storage complexity problem, in the case of FIRES.

On categorical data, we evaluate P3C against SUBCAD [9], and CLICKS [26].[4] CACTUS [10] was not included in our experiments, since CACTUS mines only a limited class of subspace clusters, and CLICKS was shown to outperform CACTUS [26].

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.

PROCLUS, HARP, SSPC, ORCLUS, and SUBCAD require the user to specify the target number of clusters. On synthetic data, we have run these 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 algorithms that require other various parameters, such as FASTDOC, MINECLUS, SSPC, and MAFIA, 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, MAFIA, and CLICKS all competing algorithms are non-deterministic; thus each of them is run 5 times, and the results are averaged. We have observed that the performance of CLICKS is very sensitive to the values of its parameters. Thus, we have run CLICKS 50 times with different parameter settings, and report the averaged results.

On the labeled real data, we have run the competing algorithms with the target number of clusters equal to the number of classes. Multiple values were tried for the other parameters required by the competing algorithms, and the results with the best accuracy are reported.

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.

[3] The code of PROCLUS, FASTDOC, HARP, SSPC, and ORCLUS was kindly provided by Kevin Yip and the project Biosphere. The code of MINECLUS was kindly provided by its authors.

[4] The code of SUBCAD and CLICKS was kindly provided by their authors.

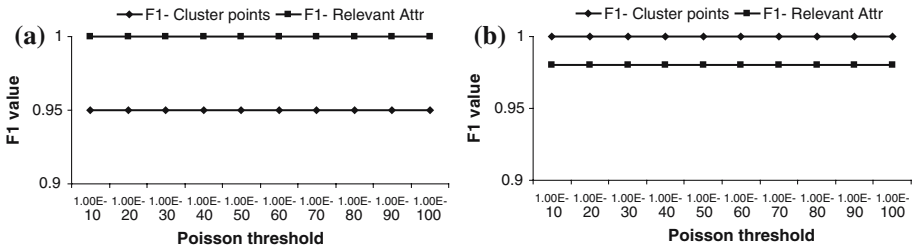

Fig. 3 P3C's sensitivity to the Poisson threshold on **a** numerical **b** categorical data sets

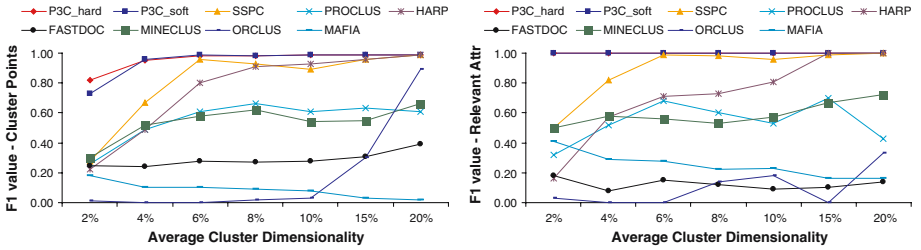

Fig. 4 Numerical data: category Uniform_Equal_Parallel

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 labeled real data, we regard class labels as cluster labels. 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 3a and 3b illustrate the accuracy of P3C_hard measured using the two $F1$_values introduced above as the Poisson threshold is progressively decreased from $1.0E-10$ to $1.0E-100$ for one of our numerical, respectively categorical, synthetic data sets. Same results are obtained for P3C_soft. We observe that P3C is remarkably robust with respect to the Poisson threshold. Similar results have been obtained on all our synthetic data sets, numerical and categorical, but are omitted due to space limitations. Consequently, the Poisson threshold can be set at any value in this range.

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 4, 5, 6, 7, 8, 9, 10, and 11 show the accuracies of the compared algorithms as a function of increasing average cluster dimensionality for the eight categories of numerical data sets. Figures 12 and 13 illustrate the same thing for the two categories of categorical data sets. We note that P3C_ hard and P3C_ soft have similar performance, with P3C_ soft

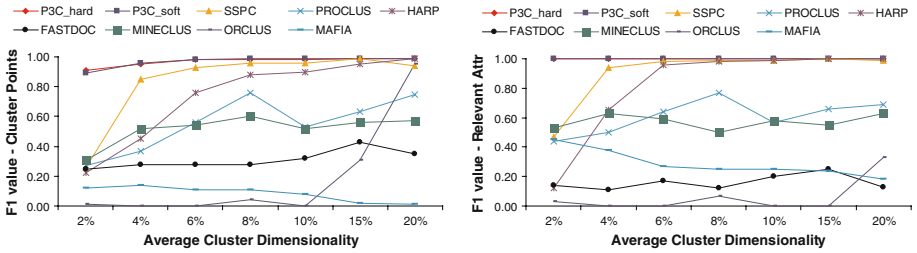

Fig. 5 Numerical data: category Uniform_Equal_NonParallel

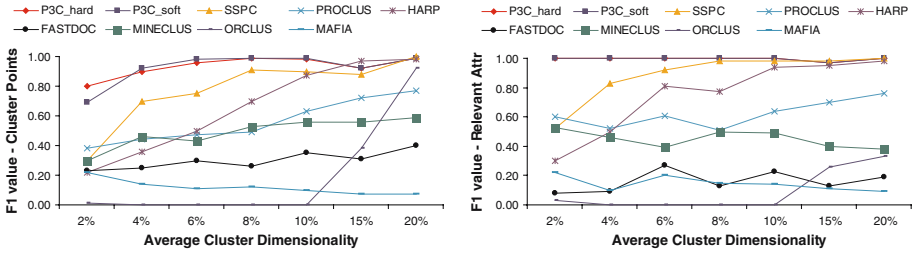

Fig. 6 Numerical data: category Normal_Equal_Parallel

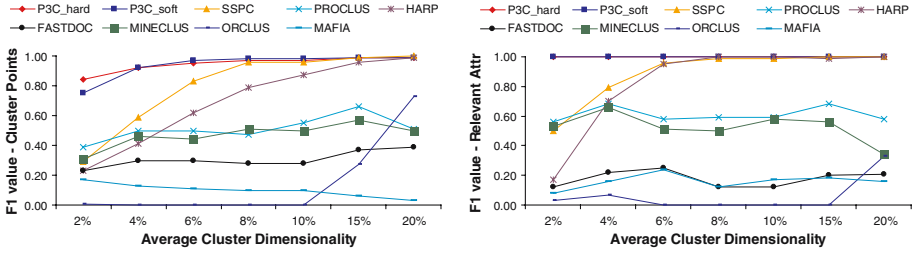

Fig. 7 Numerical data: category Normal_Equal_NonParallel

being slightly more accurate in some cases. This shows the benefit of assigning points to more than one cluster if the points satisfy the description of these clusters. We observe that P3C significantly and consistently outperforms the competing 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.

On numerical data, 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.

On numerical data, 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

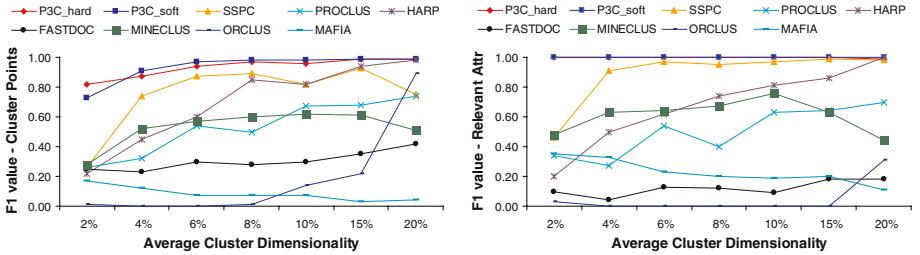

Fig. 8 Numerical data: category Uniform_Different_Parallel

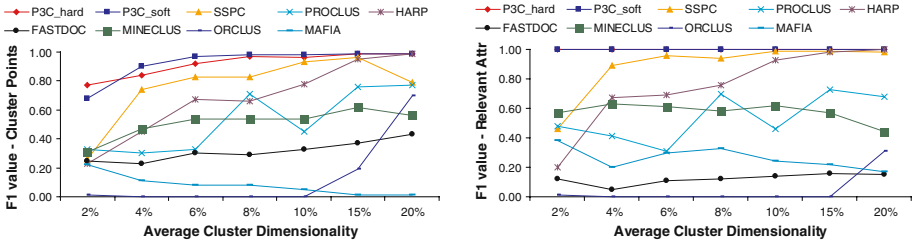

Fig. 9 Numerical data: category Uniform_Different_NonParallel

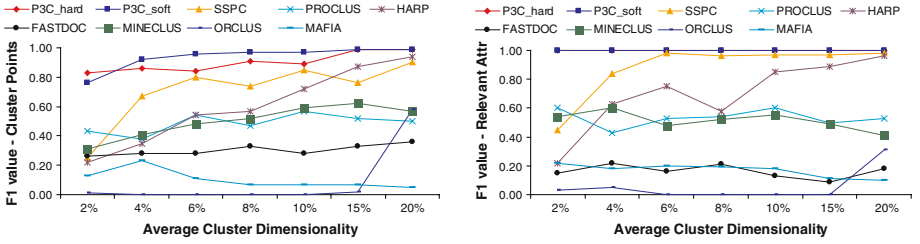

Fig. 10 Numerical data: category Normal_Different_Parallel

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 average cluster dimensionality.

We have used P3C_soft on the real data sets. On the colon cancer data, P3C discovers two 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 available implementation of MINECLUS cannot deal with such a large number of attributes. MAFIA has unacceptable long running time. The dimensionality of these two projected clusters in 11, which is much smaller than the dimensionality of the data set (i.e., 2,000). This indicates that only a relatively small fraction of genes out

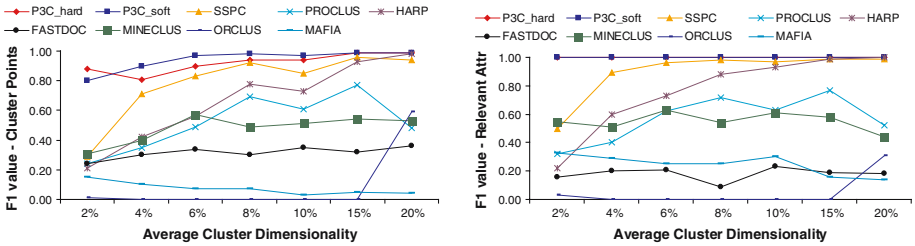

Fig. 11 Numerical data: category Normal_Different_NonParallel

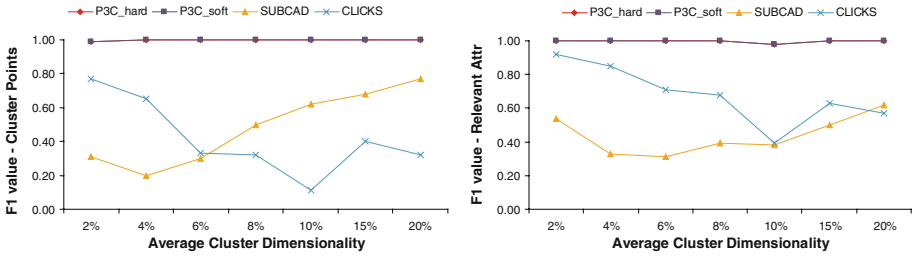

Fig. 12 Categorical data: category Equal

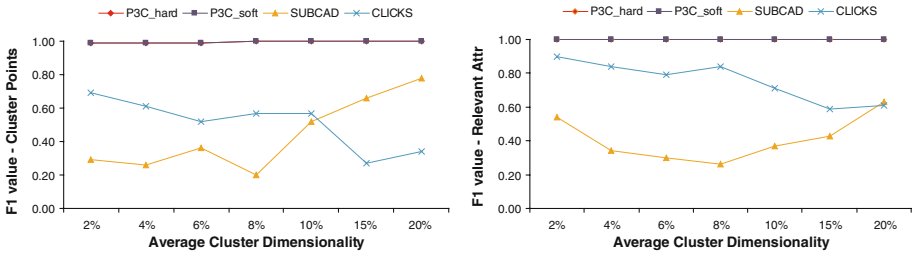

Fig. 13 Categorical data: category Different

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 two 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 two 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.

On the Hepatitis data, P3C discovers 1 projected cluster of dimensionality 2. The projected cluster corresponds to the class "live", which is much larger than the second class. The

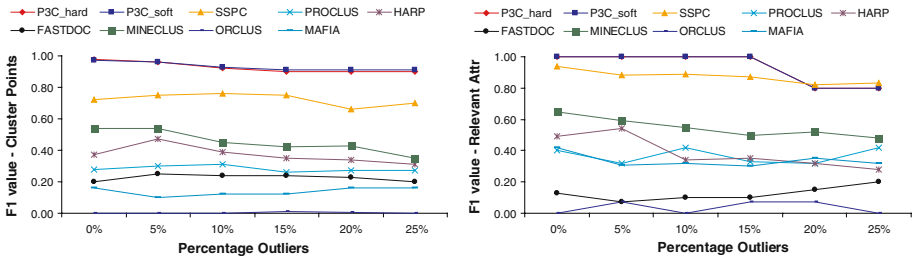

Fig. 14 Numerical data: robustness to noise

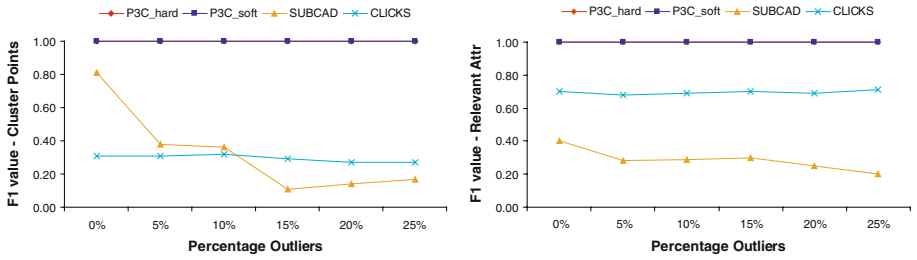

Fig. 15 Categorical data: robustness to noise

two relevant attributes are "ascites" and "varices", suggesting a possible correlation between these conditions and the outcome of the disease. P3C obtains the highest clustering accuracy (88%), followed by SUBCAD (51%), and by CLICKS (2%). Since this data set has moderate dimensionality, we have applied on it a state-of-the-art full-dimensional clustering algorithm for categorical data, LIMBO [6]. LIMBO achieves an accuracy of 61%, suggesting that the two classes are not easily discernible in full-dimensional space.

Robustness to outliers. Data sets with $n = 10,000$, $d = 100$, five clusters, average cluster dimensionality 4, and different percentages of outliers were generated. Figure 14 shows the accuracies of the compared algorithms as a function of increasing percentages of outliers for numerical data. Figure 15 illustrates the same thing on categorical data. P3C, as well as most of the competing algorithms, are robust in the presence of outliers. On numerical data, 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%. On categorical data, the accuracy of P3C is preserved as the percentage of outliers increases.

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

Figure 16a shows scalability results for numerical data sets with $d = 10$, 2 clusters, 5% outliers, average cluster dimensionality 2, and increasing database sizes. Figure 16b shows scalability results for categorical data sets with the same characteristics. The scalability of P3C with respect to database size is comparable to the scalability of the fastest competing algorithms.

Figure 17a shows scalability results for numerical data sets with $n = 10,000$, 2 clusters, 5% outliers, average cluster dimensionality 2, and increasing database dimensionality, whereas Fig. 17b illustrates the same thing for categorical data sets. P3C is relatively unaffected by increasing data dimensionality, because attributes with uniform distributions are not involved in the computation of cluster cores.

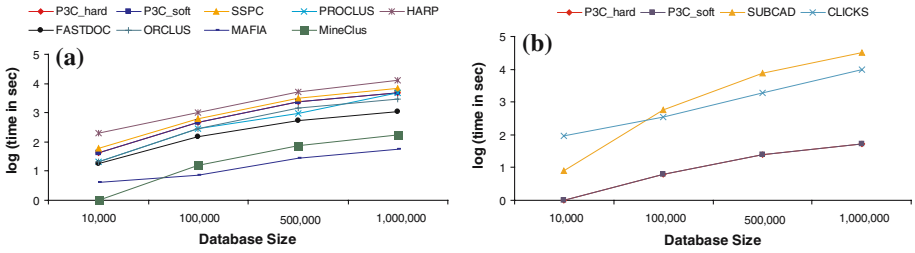

Fig. 16 Scalability with increasing database size

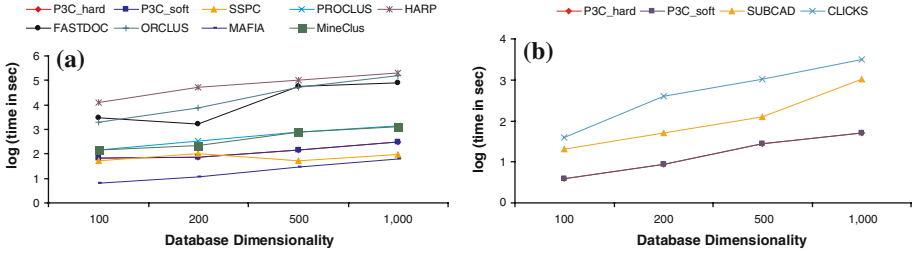

Fig. 17 Scalability with increasing database dimensionality

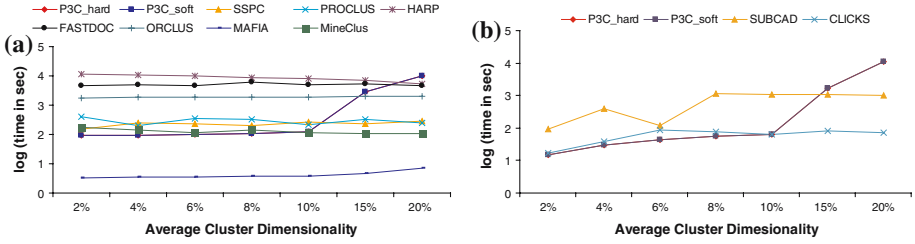

Fig. 18 Scalability with increasing average cluster dimensionality

Figure 18a shows scalability results for numerical data sets with $n = 10,000$, $d = 100$, 5 clusters, 5% outliers, and increasing average cluster dimensionality. Figure 18b illustrates similar results for categorical data sets. 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.

Figure 19 shows scalability results for categorical data sets with $n = 10,000$, $d = 100$, 2 clusters, 5% outliers, and increasing domain sizes. All algorithms are relatively unaffected by increasing domain sizes.

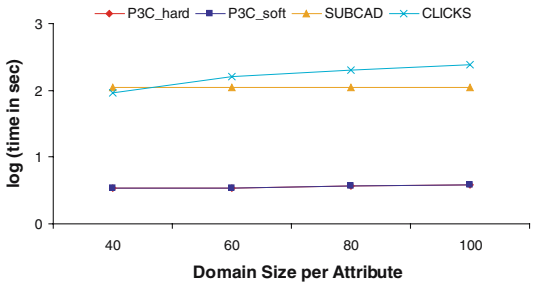

Fig. 19 Scalability with domain size per attribute

5 Related work

5.1 Projected clustering techniques

PROCLUS [1] is essentially a k-medoid algorithm adapted to projected clustering. In contrast to the standard k-medoid algorithm, initial clusters around the medoids are 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 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 [18] 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 uses the function $\mu(|C|, |D|) = |C| * (1/\beta)^{|D|}$ to measure the quality of a projected cluster, where β is a user-specified parameter. DOC computes one projected cluster at a time 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.

MINECLUS [25] improves upon DOC by proposing a deterministic method to find the optimal projected cluster around a given pivot point. Each data point is modeled as an itemset that includes the attributes in which the point is within w distance from the pivot point. The problem of finding the projected cluster around p with maximum μ value becomes the problem of mining the frequent itemset with the maximum μ value. The paper proposes a technique that modifies a known frequent pattern tree growth method used for mining frequent itemsets. Yet the accuracy of MINECLUS still depends on the three parameters α,

β, and w. To compensate for the effect of these parameters, several heuristic refinement strategies are proposed.

HARP [23] 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 the computation of initial clusters that are not necessarily reasonable approximations of real clusters. However, the relevance score of HARP makes it less effective in the case of low dimensional projected clusters.

SSPC [24] is similar in structure to PROCLUS, and its performance can be improved by the use of domain knowledge in the form of labeled objects and/or labeled attributes. SSPC uses an objective function based on the relevance score of HARP. The performance of *SSPC* depends on a user-defined parameter that controls 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 [16] 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. The performance of *EPCH* is sensitive to the values of its parameters.

We have introduced the variant of P3C for numerical data that computes disjoint clusters in [14]. In the current contribution, we extend P3C for categorical data, namely we propose novel techniques for approximating $1D$ cluster projections, and for refining cluster cores into projected clusters on categorical data. In addition, we allow P3C to compute overlapping projected clusters, which makes it comparable to subspace clustering algorithms.

5.2 Subspace clustering techniques

CLIQUE [3] overlays an axis-aligned grid over the data space by dividing each attribute into equi-length units of width ξ. A unit is *dense* if it contains more than a fraction τ of the points. A subspace cluster is defined as a maximal set of adjacent dense units in a subset of attributes. The density of a unit satisfies the downward closure property, and is used to prune effectively the search space in the process of detecting subspace clusters in all subspaces of a data set. Finally, a description is generated for each cluster by computing its cover with maximal, possibly overlapping, axis-parallel hyper-rectangles. The performance of CLIQUE is very sensitive to the resolution of the grid used. CLIQUE may miss some clusters when heuristic pruning strategy are used, or if the clusters are inadequately oriented or shaped with respect to the positioning of the grid.

MAFIA [15] is a grid-based subspace clustering algorithm that addresses some of the drawbacks of CLIQUE. MAFIA partitions each attribute into *adaptive* units that capture the data distribution on that attribute. As a consequence, the number of $1D$ dense units generated by MAFIA is much smaller than those generated by CLIQUE, which results in a

smaller search space than in CLIQUE. In addition, a cluster is represented by a cross-product of dense units, and, thus, MAFIA avoids computing clusters and their descriptions as in CLIQUE. Also, MAFIA only reports "maximal" clusters. Although MAFIA can explore more subspaces than CLIQUE in a more efficient way, it still suffers from problems similar to CLIQUE, namely sensitivity to the input parameters and the usage of a global density threshold.

SUBCLU [11] extends the formal definition of a density-connected cluster underlying the DBSCAN algorithm for the problem of subspace clustering by showing that the density-connectivity property satisfies the downward closure property. SUBCLU is able to detect subspace clusters with more general orientation and shape than grid-based approaches. However, it is also based on global density thresholds.

FIRES [13] starts with $1D$ clusters that can be obtained using any clustering algorithm of choice. These $1D$ clusters are used to construct a shared k-nearest neighbor graph: vertices correspond to $1D$ clusters, and an edge connects two vertices if each vertex is among the k-nearest neighbors of the other vertex. A modified DBSCAN algorithm is applied to this graph, and approximations of subspace clusters are generated, which are subsequently refined in a final post-processing step. Unlike other subspace clustering algorithms, FIRES does not use downward-closed properties to efficiently prune the search space. However, the performance of FIRES is very sensitive to its multiple parameters.

5.3 Categorical projected/subspace clustering techniques

SUBCAD [9] is a projected clustering algorithm for categorical data that partitions a data set into a user-specified k number of projected clusters such that a certain objective function is minimized. The performance of SUBCAD is very sensitive to its initialization performed in full-dimensional space. In addition, SUBCAD has no mechanism for detecting outliers.

CLICKS [26] computes subspace clusters as dense maximal cliques in a graph-based representation of a data set, in which the existence of edges between attribute values on different attributes is controlled by the user-defined density threshold α. Two additional post-processing steps are performed: (1) all dense maximal sub-cliques of a non-dense maximal clique are reported, and (2) maximal cliques with support larger than a user-defined σ are merged. The accuracy of CLICKS is highly dependent on the values of its parameters.

CACTUS [10] uses the same graph-based representation of a data set as CLICKS. First, CACTUS computes cluster projections on individual attributes by using the notion of a "distinguishing set" of size k, which is a set of k attribute values that uniquely occur within only one cluster. Distinguishing sets of size k are equivalent to cliques of size k in the associated graph. The assumption that clusters are uniquely identified by a core of attribute values that do not occur in other clusters is not necessarily true in all data sets. Second, cluster projections on individual attributes are used to generate cluster candidates of higher dimensionality. Due to the level-wise cluster candidates generation technique, CACTUS discovers only a limited class of subspace clusters. The accuracy of CACTUS highly depends on 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. Depending on the application needs, P3C can compute either disjoint or overlapping clusters. In contrast to previous work, P3C can be applied on both numerical and categorical data sets. Our experimental evaluation on numerous synthetic data sets and three real data sets demonstrates that P3C consistently and significantly outperforms state-of-the-art methods in terms of clustering accuracy and accuracy of the found relevant attributes; it can discover effectively very low-dimensional projected clusters while being robust to the only required parameter; it is robust to noise, and it scales well with respect to large data sets and high number of attributes.

Acknowledgments We thank anonymous reviewers for their very useful comments and suggestions. We would like to thank Kevin Yip from Yale University for providing us with the implementation of some 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. Aggarwal C, Procopiuc C, Wolf J, Yu P, and Park J (1999) Fast algorithms for projected clustering. In: Delis A, Faloutsos C, Ghandeharizadeh S (eds) Proceedings of the ACM SIGMOD international conference on management of data, Philadelphia, pp 61–72
2. Aggarwal C, Yu P (2000) Finding generalized projected clusters in high dimensional spaces. In: Chen W, Naughton J, Bernstein P (eds) Proceedings of the ACM SIGMOD international conference on management of data, Dallas, pp 70–81
3. Agrawal R, Gehrke J, Gunopulos D, Raghavan P (1998) Automatic subspace clustering of high dimensional data for data mining applications. In: Haas L, Tiwary A (eds) Proceedings of the ACM SIGMOD international conference on management of data, Seattle, pp 94–105
4. Agrawal R, Srikan R (1994) Fast algorithms for mining association rules. In: Bocca J, Jarke M, Zaniolo C (eds) Proceedings of the international conference on very large data bases VLDB, Santiago de Chile, Chile, pp 487–499
5. Alon U, Barkai N, Notterman D, Gish K, Ybarra S, Mack D, Levine A (1999) Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays. Proc Natl Acad Sci USA 96(12):6745–6750
6. Andritsos P, Tsaparas P, Miller J, Sevcik K (2004) LIMBO: scalable clustering of categorical data. In Proceedings of international conference on extending database technology EDBT, Heraklion, Greece, pp 123–146
7. Beyer K, Goldstein J, Ramakrishnan R, Shaft U (1999) When is nearest neighbor meaningful? Lecture Notes in Computer Science, vol. 1540. Springer, Berlin, pp 217–235
8. Dempster A, Laird N, Rubin D (1977) Maximum likelihood for incomplete data via the EM algorithm. J Roy Stat Soc 39:1–38
9. Gan G, Wu J (2004) Subspace clustering for high dimensional categorical data. ACM SIGKDD Explor Newslett 6(2):87–94
10. Ganti V, Gehrke J, Ramakrishnan R (1999) CACTUS — clustering categorical data using summaries. In: ACM SIGKDD international conference on knowledge discovery and data mining, San Diego, pp 73–83
11. Hinneburg A, Keim D (2003) A general approach to clustering in large databases with noise. Knowl Inf Syst 5(4):387–415
12. Kailing K, Kriegel H, Krőger P (2004) Density-connected subspace clustering for high-dimensional data. In: Berry M, Dayal U, Kamath C, Skilicorn D (eds) Proceedings of the SIAM international conference on data mining, Lake Buena Vista, April 2004, pp 1–11
13. Kriegel H, Krőger P, Renz M, Wurst S (2005) A generic framework for efficient subspace clustering of high-dimensional data. In: Proceedings of the IEEE ICDM international conference on data mining, Houston, pp 250–257
14. Moise G, Sander J, Ester M (2006) P3C: a robust projected clustering algorithm. In: Proceedings of the IEEE ICDM international conference on data mining, Hong Kong, pp 414–425

15. Nagesh H, Goil S, Choudhary A (2001) Adaptive grids for clustering massive data sets. In: Proceedings of the SIAM international conference on data mining, Chicago, pp 1–17
16. Ng K, Fu A, Wong C (2005) Projective clustering by histograms. IEEE Trans Knowl Data Eng 17(3):369–383
17. Parsons L, Haque E, Liu H (2004) Subspace clustering for high dimensional data: a review. ACM SIGKDD Explor Newslett 6(1):90–105
18. Procopiuc C, Jones M, Agarwal P, Murali T (2002) A Monte Carlo algorithm for fast projective clustering. In: Franklin M, Moon B, Ailamaki A (eds) Proceedings of the ACM SIGMOD international conference on management of data, Madison, pp 418–427
19. Rousseeuw P, Van Zomeren B (1990) Unmasking multivariate outliers and leverage points. J Am Stat Assoc 85(411):633–651
20. Snedecor G, Cochran W (1989) Statistical methods. Iowa State University Press, Cambridge
21. Tang J, Chen J, Fu A, Cheung W (2007) Capabilities of outlier detection schemes in large datasets, framework and methodologies. Knowl Inf Syst 11(1):45–84
22. Wang J, Karypis G (2006) On efficiently summarizing categorical databases. Knowl Inf Syst 9(1):19–37
23. Yip K, Cheung D, Ng M (2004) HARP: a practical projected clustering algorithm. IEEE Trans Knowl Data Eng 16(11):1387–1397
24. Yip K, Cheung D, Ng M (2005) On discovery of extremely low-dimensional clusters using semi-supervised projected clustering. In: Proceedings of the IEEE ICDE international conference on data engineering, Tokyo, pp 329–340
25. Yiu M, Mamoulis N (2005) Frequent-pattern based iterative projected clustering. IEEE Trans Knowl Data Eng 17(2):176–189
26. Zaki M, Peters M, Assent I, Seidl T (2005) CLICKS: an effective algorithm for mining subspace clusters in categorical datasets. In: Grossman R, Bayardo R, Bennett K (eds) Proceedings of ACM SIGKDD international conference on knowledge discovery and data mining, Chicago, pp 733–742

Authors Biography

Gabriela Moise received her MSc in Computer Science from the University of Alberta, Canada in 2003. She is currently a PhD student in the Department of Computing Science at the University of Alberta, Canada. Her research interests include data mining, databases, bioinformatics, and machine learning.

Jörg Sander received his MS in Computer Science in 1996, and his PhD in Computer Science in 1998, both from the University of Munich, Germany. He worked one year as a post-doctoral fellow at the University of British Columbia, Canada, and joined the University of Alberta, Canada, in 2001 as an Assistant Professor. His research interests include Knowledge Discovery in Databases, especially clustering and data mining—with particular interest in spatial, spatio-temporal and biological applications, and techniques and index structures that improve scalability of basic operations such as similarity search in large databases (Current information can be found at http://www.cs.ualberta.ca/ joerg.)

Martin Ester received a PhD in Computer Science from the Swiss Federal Institute of Technology (ETH Zurich) in 1990 with a thesis in the area of knowledge-based systems. For four years, he worked for Swissair's New Technologies and Expert Systems group as a Senior and Advisory Systems Engineer. In 1993, he returned to academia and joined the Computer Science Department of University of Munich (LMU) as an Assistant Professor. Since 2001, he has been an Associate Professor at the School of Computing Science of Simon Fraser University, where he co-directs the database and data mining lab. He regularly serves on the program committees of the premier conferences in his field and is an Associate Editor of Knowledge and Information Systems. Martin Ester's research interests are in the areas of databases and data mining, particularly clustering algorithms, multi-relational clustering and classification, mining biological databases and preference queries.