# Efficient Clustering of Uncertain Data

Wang Kay Ngai, Ben Kao, Chun Kit Chui
Dept. of Computer Science
The University of Hong Kong
wkngai,kao,ckchui@cs.hku.hk

Reynold Cheng
Department of Computing
Hong Kong Polytechnic University
csckcheng@comp.polyu.edu.hk

Michael Chau
School of Business
The University of Hong Kong
mchau@business.hku.hk

Kevin Y. Yip
Dept. of Computer Science
Yale University
yuklap.yip@yale.edu

## Abstract

*We study the problem of clustering data objects whose locations are uncertain. A data object is represented by an uncertainty region over which a probability density function (pdf) is defined. One method to cluster uncertain objects of this sort is to apply the UK-means algorithm, which is based on the traditional K-means algorithm. In UK-means, an object is assigned to the cluster whose representative has the smallest expected distance to the object. For arbitrary pdf, calculating the expected distance between an object and a cluster representative requires expensive integration computation. We study various pruning methods to avoid such expensive expected distance calculation.*

## 1. Introduction

In many applications, data contains inherent uncertainty. A number of factors contribute to the uncertainty such as the random nature of the physical data generation and collection process, measurement error, and data staling. As an example, consider the problem of clustering mobile devices continuously according to the periodic updates of their locations. One application of the clustering is the selection of a device as the leader for each cluster. A leader's role is to collect data (such as location data) from its cluster members and to communicate with a server or a base station with batched updates. In this way, most communications are short-ranged messages between cluster members and their leaders. In this example, we observe that data, as received by a server, contains the following type of uncertainty:

- The physical devices that determine vehicle locations are accurate only up to a certain precision.
- The current locations can only be estimated based on the last reported values. That is, data is always stale.

As another example, an animal-tracking device might report the location of an animal periodically. One can compile the location data into a histogram that indicates the relative probability of the animal being located at a particular region. The whereabouts of the animal can then be represented by a probability-density function (pdf). Applying clustering on the data can reveal interesting insights on the relationship and social behaviors of the animals.

In this paper we study the problem of clustering objects with *multi-dimensional uncertainty*. In particular, an object is not a simple point in space, but is represented by an uncertainty region over which a pdf is defined. Formally, we consider a set of $n$ objects $o_i, 1 \leq i \leq n$ in an $m$-dimensional space. Object $o_i$ is represented by a pdf $f_i : \mathbb{R}^m \rightarrow \mathbb{R}$ that specifies the density of each possible location of object $o_i$. The methods to be discussed in this paper do not rely on any special forms of $f_i$, but only require that for each object $o_i$, there is a finite region $A_i$ such that $\forall x \notin A_i, f_i(x) = 0$. This requirement allows each object to be approximated by a finite bounding box.

The goal of clustering is to group the objects into $k$ clusters, such that the total expected distance from the objects to a representative point of their corresponding clusters is minimized. Going back to our mobile-device example again, if peer-to-peer transmission cost is proportional to the transmission distance, then the above goal is equivalent to minimizing the expected transmission cost. So, suppose $c_i$ represents the cluster to which object $o_i$ belongs, and $p_{c_i}$ is the representative point of cluster $c_i$, we want to find the values of $p_{c_i}$'s and $c_i$'s such that the objective function

This research is supported by Hong Kong Research Grants Council grant HKU 7134/06E.

$\sum_{i=1}^{n} ED(o_i, p_{c_i}) = \sum_{i=1}^{n}(\int f_i(x) d(x, p_{c_i}) dx)$ is minimized, where $ED$ is the expected distance based on a metric $d$ (e.g., Euclidean distance).

In real applications, there are practical ways to specify the pdfs. For example, measurement errors can be modeled by Gaussian distributions with the means and variances estimated from sample data with known real locations obtained by some external means [19]. Also, uncertainty due to data staling can be modeled according to the last reported location and velocity and the vehicle's properties (e.g., maximum speed) [19]. Assuming the pdfs $f_i$ to be non-zero only over a finite region is thus reasonable since the density functions drop super-linearly (e.g., exponentially for Gaussian distributions) so that the locations outside a certain region have negligible contribution to expected distances.

We have discussed in a separate study the importance of considering data uncertainty explicitly in clustering with a focus on the quality of the clustering results [2]. And an algorithm called UK-means was proposed. The algorithm was applied to moving object uncertainty and was shown to improve accuracy of the clusters formed. Uniform distribution was assumed for the uncertainty associated with the data used. While the method can be generalized to other distributions, there are significant efficiency issues that have not been addressed. This paper, however, focuses on efficiency issues. Given a fixed clustering process, we study ways to reduce the running time. Efficiency is of particular importance in real time applications such as the one about mobile devices. It is also important when a large amount of object data is involved.

Traditional clustering process often requires the definition of a distance metric. For example, in K-means clustering [14], an object $o$ is assigned to a cluster $c$ such that the *distance* between $o$ and a representative of $c$ is the smallest among all clusters. We remark that with multi-dimensional uncertainty, an object is no longer a single point in space but is represented by a pdf over an uncertainty region. The definition of *distance* thus has to be revisited. In this paper we choose to use *expected distance* as the distance measure. As we will explain, the expected distance metric not only provides an intuitive way of handling uncertainty, but also enables the development of efficient clustering algorithms. For arbitrary pdfs, computing the expected distance of an uncertain object and a cluster representative requires very expensive integration operations, which are often hundreds or even thousands of times more expensive than a simple distance computation. As we will discuss later in this paper, traditional clustering techniques have to be refined so that the algorithms become computationally feasible. One of the major contributions of this paper is about pruning techniques that can significantly reduce the number of expected distance calculations in the clustering process.

The rest of this paper is organized as follows. Section 2 discusses some related work on uncertain data mining and in particular uncertain data clustering. We discuss some issues that are not dealt with in previous studies and how we tackle them. Section 3 describes a basic clustering algorithm for uncertain data based on K-means. We discuss the performance bottleneck of the algorithm, and describe a simple method for speeding up the clustering process. In Sections 4 and 5 we propose four methods for further improving performance. Section 6 demonstrates the effectiveness of the methods by extensive experiments. Section 7 discusses some observations and concludes the study.

## 2. Related work

There has been significant research interest in data uncertainty management in recent years. Data uncertainty has been broadly classified into two types. The first type concerns existential uncertainty. For example, a tuple in a relational database could be associated with a probability value that indicates the confidence of its presence [16]. This "probabilistic database model" has been applied to semi-structured data and XML [10]. The second type concerns value uncertainty. Under this type, a data item is modeled as a closed region which bounds its possible values, together with a pdf of its value [3, 19]. This model can be used to quantify the imprecision of location and sensor data in a constantly evolving environment, as in our moving vehicles example. For value uncertainty, most work has been devoted to "imprecise queries," which provide probabilistic guarantees over correctness of answers. For example, in [4], indexing solutions for range queries over uncertain data have been proposed. The same authors also proposed solutions for aggregate queries such as nearest-neighbor queries in [3]. Notice that all these works have applied the study of uncertain data management to simple database queries, instead of to the relatively more complicated data analysis and mining problems.

Data clustering is one of the most studied areas in data mining research. Depending on application, there are several goals of clustering: to identify the (locally) most probable values of the model parameters [5] (e.g., means of Gaussian mixtures), to minimize a certain cost function (e.g. the total within-cluster squared distance to centroid [14]), or to identify high-density connected regions [8] (e.g., areas with high population density). The current study falls into the second category.

On the topic of clustering uncertain data, Hamdan and Govaert have addressed the problem of fitting mixture densities to uncertain data for clustering using a modified Expectation-Maximization (EM) algorithm [9]. They supposed that data observed were the sampling results from a distribution mixture and aimed to find the maximum likelihood estimation of the mixture model parameters. The algorithm is customized for EM and thus cannot be readily applied to other clustering situations. Clustering on interval data has also been studied. Different distance measures, like city-block distance or Minkowski distance, have been used in measuring the similarity between two intervals [11]. The pdfs of the intervals are not taken into account in most of these metrics. Another related area of research is fuzzy clustering, which has been long studied in fuzzy logic [17]. In fuzzy clustering, a cluster is represented by a fuzzy subset of a set of objects. Each object has a "degree of belongingness" for each cluster. In other words, an object can belong to more than one cluster, each with a different degree. The fuzzy c-means algorithm is one of the most widely used fuzzy clustering methods [6]. Different fuzzy clustering methods have been applied on normal or fuzzy data to produce fuzzy clusters [18]. While their work focuses on creating fuzzy clusters (i.e., each object can belong to more than one cluster with different degrees), our work is developed for hard clustering based on the uncertainty model of objects, in which each object can only belong to one cluster.

Recently, there have been studies on density-based clustering of uncertain data. The FDBSCAN [12] and FOPTICS [13] algorithms are based on DBSCAN [8] and OPTICS [1], respectively. Instead of identifying regions with high density, these algorithms identify regions with high expected density, based on the pdfs of the objects.

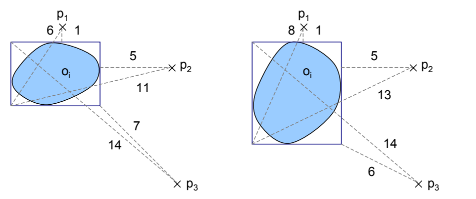
## 3. Basic algorithm and min-max-dist pruning

In this section we describe two algorithms for clustering uncertain data. The first one is called UK-means (which stands for Uncertain K-means) [2]. UK-means basically follows the well-known K-means algorithm except that it uses *expected distance* when determining which cluster an object should be assigned to. The second algorithm uses the idea of min-max distance pruning in UK-means with the objective of reducing the number of expected distance calculations.

UK-means starts by randomly selecting $k$ points as cluster representatives. Each object $o_i$ is then assigned to the cluster whose representative $p_j$ has the smallest expected distance from $o_i$ ($ED(o_i, p_j)$) among all clusters. After the assignment, cluster representatives are recomputed as the mean of the centers of mass of the assigned objects. The two steps form an iteration, which is repeated until the convergence of the objective function.

Computing the expected distance, $ED(o_i, p_j)$, requires the computation of the integral $\int f_i(x) d(x, p_j) dx$, where $f_i(x)$ is the probability density of a point $x$ in the uncertainty region of $o_i$, and $d(x, p_j)$ is the distance between $x$ and $p_j$. In practice, the pdf is approximated by dividing the uncertainty region into a number of grids. The probability density of a sample point in each grid is recorded. To calculate the integral, we calculate the distance of each sample to $p_j$, and then approximate the integral by finding the sum of the distances, weighted by the corresponding probability density of the sample points. For accuracy, thousands of samples are needed. Expected distance calculation is thus a computationally expensive operation.

Not only is expected distance expensive to compute, it is also one of the most frequently executed operations. This is because an expected distance is calculated between each object and cluster representative pair for each iteration. That is to say, if UK-means iterates $t$ times for a set of $n$ objects to form $k$ clusters, UK-means would compute a total of $nkt$ expected distances.



(a) An object with a small uncertain region. (b) An object with a large uncertain region.

**Figure 1. The min-max-dist pruning method.**

To improve efficiency, a pruning technique based on the concept of min-max distance can be used to avoid unnecessary expected distance calculations. The basic idea is to use inexpensive distance calculations to identify cluster representatives that cannot be the closest one to an object. Hence, the expected distances between those representatives and the object need not be computed. More specifically, for each object $o_i$, we define a minimum bounding rectangle (MBR) outside which the object has zero (or negligible) probability of occurrence. Now, for each cluster representative $p_j$, we compute the minimum distance ($MinDist_{ij}$) and maximum distance ($MaxDist_{ij}$) between $p_j$ and the MBR of $o_i$ (see Figure 1a). Among all the maximum distances, the smallest one is called the min-max distance $\hat{d}_i$ between $o_i$ and the cluster representatives. For example, in Figure 1a, we have $MinDist_{i1}$ = 1, $MinDist_{i2}$ = 5, $MinDist_{i3}$ = 7, $MaxDist_{i1}$ = 6, $MaxDist_{i2}$ = 11 and $MaxDist_{i3}$ = 14. Since $p_1$ gives the smallest maximum distance, we have $\hat{d}_i = 6$. One can show that any cluster representative $p_j$ whose minimum distance $MinDist_{ij}$ is larger than $\hat{d}_i$ cannot be the one with the smallest expected distance from $o_i$. That is,

$$ED(o_i, p_j) \geq ED(o_i, p_{j^*}), \quad (1)$$

where $p_{j^*}$ is a cluster representative with the smallest maximum distance (i.e., $MaxDist_{ij*} = \hat{d}_i$). For example, in Figure 1a, since $MinDist_{i3} = 7$, which is larger than $\hat{d}_i = 6$, $p_3$ cannot be the representative closest to $o_i$ and thus the expected distance $ED(o_i, p_3)$ needs not be computed.

For those cluster representatives that cannot be pruned, their expected distances from object $o_i$ are calculated. Object $o_i$ is then assigned to the one with the smallest expected distance. Note that during this process, whenever an expected distance $ED(o_i, p_j)$ of a cluster representative $p_j$ is computed, we can refine $\hat{d}_i$ to $\min(\hat{d}_i, ED(o_i, p_j))$. Any remaining cluster representative $p_{j'}$ can be pruned if $MinDist_{ij'} > \hat{d}_i$. This potentially reduces the number of expected distance calculations further. We call the above method the min-max-dist pruning method.

The min-max-dist pruning method is effective in saving expected distance calculations of cluster representatives that are much farther away from the closest one. However, it suffers when the MBR gives poor distance estimates. This occurs when the uncertainty region of an object is large. For example, in Figure 1b, the maximum distance of $p_1$ is increased to 8 due to a larger MBR. The min-max distance $\hat{d}_i$ is now 8. This new min-max distance is no longer smaller than $MinDist_{i3}$, and thus representative $p_3$ cannot be pruned and $ED(o_i, p_3)$ needs to be computed.

Note that the pruning effectiveness of min-max-dist pruning relies on the $MinDist$ and the $MaxDist$ bounds. The closer they are to the true expected distance, the more effective is the pruning. We will describe several methods for achieving tighter bounds in the following two sections.

## 4. Improving distance bounds

In [15], the triangle inequality is applied to derive an upper and lower bound of a distance between two points in order to reduce many distance computations between data points in a hierarchical clustering process of conventional data. In this section we extend such idea to derive a good upper and lower bound of an expected distance between a data object and a cluster representative point in order to reduce expected distance computations in UK-means. Let $y$ be a point called an *anchor point*. Since the distance function $d$ is a metric, by the triangle inequality, we can derive the following upper bound for the expected distance between an object $o_i$ and a cluster representative $p_j$:

$$ED(o_i, p_j) \leq ED(o_i, y) + d(y, p_j). \qquad (2)$$

If the expected distance $ED(o_i, y)$ between the object $o_i$ and the anchor point $y$ is pre-computed, then by Inequality 2, an upper bound of the expected distance $ED(o_i, p_j)$ can be computed using only one inexpensive distance calculation between $y$ and $p_j$. We call this upper bound estimation the $U_{pre}$ method (for "Upper bound estimation based on Pre-computed Expected Distances").

Method $U_{pre}$ can be integrated into the min-max-dist pruning strategy easily. Recall that with the basic min-max-dist pruning, we estimate an upper bound of $ED(o_i, p_j)$ by finding the maximum distance between a cluster representative $p_j$ and the MBR of an object $o_i$. Let us call this bound $U_{MBR,ij}$. With $U_{pre}$, we simply compare the upper bound as determined by Inequality 2 with $U_{MBR,ij}$ and take the smaller one as $MaxDist_{ij}$. The rest of min-max-dist pruning is the same as what we have described in Section 3.

We note that since the upper bound from Inequality 2 is used as $MaxDist_{ij}$ only if it gives a tighter bound, $\hat{d}_i$ is always at least as good (i.e., small) as the value obtained without using the $U_{pre}$ method, and is potentially better. As a consequence, potentially more cluster representatives will have their $MinDist$'s $> \hat{d}_i$ and thus more of them can be pruned, leading to a more efficient algorithm.

Moreover, the use of $U_{pre}$ is not restricted to one anchor point $y$. One can use multiple anchor points and pick the one that gives the best $MaxDist_{ij}$ upper bound. The trade-off is that the more anchor points used, the tighter is the bound (and hence a higher pruning potential) at the expense of a higher pre-computation cost. We will elaborate on this point further later in this section.

Similarly, a lower bound for $ED(o_i, p_j)$ can be derived:

$$ED(o_i, p_j) \geq |d(y, p_j) - ED(o_i, y)| \qquad (3)$$

Again, if the expected distance $ED(o_i, y)$ between object $o_i$ and anchor point $y$ is pre-computed, then a lower bound of $ED(o_i, p_j)$ can be determined by an inexpensive computation of $d(y, p_j)$. We call this lower bound estimation method $L_{pre}$. Method $L_{pre}$ can be incorporated into min-max-dist pruning in a way similar to how we use Method $U_{pre}$. When determining $MinDist_{ij}$ of a cluster representative $p_j$ and an object $o_i$, we compare the lower bound $|d(y, p_j) - ED(o_i, y)|$ against the minimum distance between $p_j$ and the MBR of $o_i$. The larger value is taken as $MinDist_{ij}$. Since the lower bound $MinDist_{ij}$ is potentially tighter (i.e., larger), a cluster representative $p_j$ is more likely be pruned due to $MinDist_{ij} > \hat{d}_i$.

While Methods $U_{pre}$ and $L_{pre}$ can potentially increase the pruning power, they come with a cost. If $r$ anchor points are used, a total of $nr$ expected distances need to be pre-computed. The two methods would induce an overall saving only if the total number of additional representatives being pruned in all iterations is larger than $nr$. It is therefore important to choose a set of anchor points that has sufficient pruning power, but is small enough to avoid large overhead.

Let us revisit the upper bound derived by Method $U_{pre}$ (Inequality 2). To make the upper bound as tight as possible, we want both $ED(o_i, y)$ and $d(y, p_j)$ to be small. We now show that the anchor point $y$ that minimizes $ED(o_i, y)$ must lie inside the MBR of $o_i$. Suppose that a point $y$ is outside the MBR of $o_i$ and $y^*$ is the point on the boundary of the MBR that is closest to $y$. There are two possible cases, either the line $y^*y$ is perpendicular to an edge of the MBR (Figure 2a), or $y^*$ is at a corner of the MBR (Figure 2b). Consider any point $x$ inside the uncertainty region of $o_i$. We observe that if $x$ lies on the line $y^*y$ (i.e., $x = z$), then $d(x,y) > d(x,y^*)$; Otherwise, $d(x,y) = \frac{d(x,z)}{\cos \angle yxz} > \frac{d(x,z)}{\cos \angle y^*xz} = d(x,y^*)$. Therefore in all cases, $ED(o_i,y) = \int f_i(x)d(x,y)dx > \int f_i(x)d(x,y^*)dx = ED(o_i,y^*)$, which proves that the point $y$ that minimizes $ED(o_i,y)$ must be within the MBR of $o_i$.

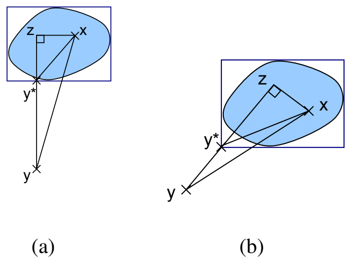


**Figure 2. Proving that the point minimizing the expected distance to an object must be within its MBR.**

Unfortunately, for a general pdf, there are no simple ways to compute the exact location of the point that minimizes the expected distance, so a reasonable scheme is to pick various anchor points within the MBR of $o_i$. Another consideration is that we also want $d(y,p_j)$ to be small. However, the location of $p_j$ is not known in advance at the time when the anchor points are picked and their expected distances from $o_i$ are pre-computed. A reasonable option is thus to pick anchor points on the different sides of the object's MBR. Depending on the number of anchor points allowed, we have chosen several reasonable sets of anchor points. In a one-point scheme, the center of the MBR is picked. In a five-point scheme, the center of the four faces of the MBR are also used as it is guaranteed that for any representative outside the MBR, at least one of the four points would be closer to the representative than the center does. In a nine-point scheme, the four corners of the MBR are also used to cater for cases in which the representative is closer to a corner than to the four faces. Figure 3 illustrates the choice of anchor points.

The above schemes are also reasonable choices for Method $L_{pre}$. This is because when a cluster representative $p_j$ is outside the MBR of $o_i$, we want $ED(o_i,y)$ to be small and $d(y,p_j)$ to be large (see Inequality 3). The former goal is achieved by picking anchor points within the MBR of $o_i$, and the latter goal is achieved by trying anchor points

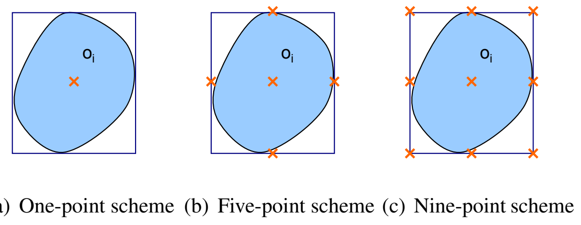


**Figure 3. Anchor point selection schemes.**

at different sides of the MBR. In Section 6 we will show empirically the cost and benefits of the various schemes of anchor point selection.

Finally, we remark that Methods $U_{pre}$ and $L_{pre}$ are especially useful in applications for which some of the objects change their pdfs and/or locations while some others do not, and that updates of the clustering results are needed periodically. In this scenario, the cost of pre-computing the expected distances of static objects can be amortized over the multiple executions of the clustering algorithm. Some of the pre-computation overhead can thus be ignored.

## 5. Reusing expected distance calculations

Let us consider Inequality 2 again. In Section 4, we discussed why we picked anchor points within the MBR of an object $o_i$. The idea was to make the term $ED(o_i,y)$ as small as possible so that the upper bound was tight. Inequality 2 suggests that another way of making the bound tight is to make $d(y,p_j)$ as small as possible. That is to say, we want an anchor point to be as close to the cluster representative $p_j$ as possible. Now the question is which point $y$ is close to $p_j$ while its expected distance to $o_i$, namely, $ED(o_i,y)$ is readily available? A reasonable answer is the location of the representative of cluster $j$ in the previous iteration of the clustering process. Such idea is used in [7] to derive a lower bound of a distance between a data point and a cluster representative point in K-means clustering of conventional data. In this section we extend the idea to derive both upper and lower bounds of the expected distances.

Consider a cluster $j$ whose representative is $p_j$ during a particular iteration of the clustering process. Let $p'_j$ be the updated representative of cluster $j$ in the next iteration. We note that the distance between $p_j$ and $p'_j$, i.e., $d(p_j,p'_j)$ is likely to be small. This is especially true during the later iterations of the clustering process when cluster representatives shift by small distances only.

Given an object $o_i$, recall that min-max-dist pruning determines if a cluster representative $p_j$ could be pruned by comparing $MinDist_{ij}$ against the min-max-dist threshold

$\hat{d}_i$. There are two cases:

1. $MinDist_{ij} > \hat{d}_i$. In this case, $p_j$ is pruned and $ED(o_i, p_j)$ is not calculated. In the next iteration, we compare $MinDist_{ij'}$ against (an updated but likely similar) $\hat{d}_i$. Since $p_j$ and $p'_j$ are likely to be close to each other, $MinDist_{ij'}$ is likely to be similar in value to $MinDist_{ij}$. Hence, $p'_j$ is likely to be pruned by min-max-dist pruning in the next iteration.

2. $MinDist_{ij} \leq \hat{d}_i$. In this case, $p_j$ cannot be pruned and min-max-dist pruning will calculate $ED(o_i, p_j)$. In the next iteration, min-max-dist pruning needs an upper bound of $ED(o_i, p'_j)$. This upper bound can be easily obtained by taking $p_j$ as the anchor point for Inequality 2.

For Case (2), triangle inequality gives

$$ED(o_i, p'_j) \leq ED(o_i, p_j) + d(p_j, p'_j) \tag{4}$$

Since $ED(o_i, p_j)$ was calculated in the previous iteration, an upper bound of $ED(o_i, p'_j)$ can be obtained by an inexpensive distance calculation $d(p_j, p'_j)$. We call this method $U_{cs}$, which stands for "upper bound estimation based on Cluster Shift."

Likewise, a good lower bound of $ED(o_i, p'_j)$ can also be obtained by taking $p_j$ as the anchor point:

$$ED(o_i, p'_j) \geq |ED(o_i, p_j) - d(p_j, p'_j)| \tag{5}$$

We call this method $L_{cs}$.

A big advantage of Methods $U_{cs}$ and $L_{cs}$ is that they require no pre-computation of expected distances that was needed by Methods $U_{pre}$ and $L_{pre}$.

The four methods $U_{pre}$, $L_{pre}$, $U_{cs}$ and $L_{cs}$ are all independent of each other. One can choose to apply any combination of the four methods. For notational convenience, we concatenate the methods' names to represent strategies that combine a number of such methods. So, for example, $U_{pre}L_{pre}L_{cs}$ represents a method that combines Method $U_{pre}$, Method $L_{pre}$ and Method $L_{cs}$ together. In general, $U_{pre}$ and $L_{pre}$ provide effective pruning during early iterations of the clustering process when a relatively large number of objects are still migrating among multiple clusters. Methods $U_{cs}$ and $L_{cs}$, on the other hand, give significant pruning during late iterations when cluster representatives shift by only small distances across iterations.

## 6. Experiments

As we have explained, expected distance calculations are the performance bottleneck of the algorithms. Therefore, we measure the effectiveness of the pruning methods based on the average number of expected distance calculated per object per iteration in a clustering process. We denote this number by $N_{ED}$. Note that under the brute-force implementation of UK-means, during each iteration, the algorithm calculates the expected distance of an object to every single cluster representative. Therefore, $N_{ED} = k$ for the brute-force algorithm, where $k$ is the number of clusters. The value $k$ is thus the baseline reference when we discuss the effectiveness of the other methods.

**Table 1. Parameters and baseline values.**

| Parameter | Description | Value |
|---|---|---|
| $n$ | number of objects | 20,000 |
| $k$ | number of clusters | 49 |
| $d$ | maximum length of an MBR's side | 10 |
| $s$ | number of sample points | 196 |

### 6.1. Settings

In our experiment, we evaluate the pruning methods on two types of datasets: one generated with intrinsic cluster patterns, the other without. As we will see later, the general observations that we draw about the relative performance of the pruning methods do not differ significantly across these two models. In this paper we focus on the latter case.

Data generation follows the following procedure. All objects are to be located in a $100 \times 100$ 2D space. Each object is first represented by an MBR with random side lengths. For datasets without cluster patterns, the MBRs are simply randomly positioned in the space. For datasets with cluster patterns, $k$ points are chosen randomly as the cluster centers such that the distance between any two of them is at least $\frac{100}{2\sqrt{k}}$. Then the MBRs of the objects are divided into $k$ groups, each assigned to one cluster center. For each MBR assigned to a cluster center, its center position is randomly chosen from all the positions within a distance of $\frac{100}{\sqrt{k}}$ from the cluster center. In this way, each object assigned to a cluster is likely to be closer to the center of its cluster than to those of any other clusters, hence producing some natural cluster patterns.

During clustering, expected distances are computed based on discrete sample points of the pdfs. Therefore, the pdf of an object is specified by the probabilities at a finite number of discrete sample points in its MBR. For each object, we divide its MBR into a $\sqrt{s} \times \sqrt{s}$ grid, where $s$ is the number of samples used to approximate the pdf. A probability is randomly generated for each cell of the grid. The cell probabilities are then normalized so that they sum to 1.

In our experiments we study how the following factors affect the algorithm's performance, namely, $n$: the number of objects to be clustered; $k$: the number of clusters; $d$: the maximum length of a side of an MBR; and $s$: the number of sample points used in representing an object's pdf. Table 6.1 shows the baseline setting of these parameters. In the experiments, we perform sensitivity study on these parameters, varying the value of one parameter at a time.

For datasets without cluster patterns, the initial cluster representatives are picked uniformly from the 2D space. For datasets with cluster patterns, the initial cluster representatives are set as the centers of mass of some randomly chosen objects.

For each set of parameter values, the clustering process is executed 50 times and the average result is reported. We compare $N_{ED}$ of the brute-force UK-means algorithm, the basic min-max-dist pruning algorithm, and various combinations of our four pruning strategies.

All our codes are written in Java 1.5. The experiments are run on Windows machines with an Intel 3.2GHz Pentium 4 processor and 1024MB of memory.

## 6.2. Results

We first investigate the effectiveness of the pruning methods with different dataset sizes ($n$). Figure 4 shows $N_{ED}$ under different strategies as $n$ varies from 1,000 to 30,000 objects. The data objects in this experiment are generated without cluster patterns.

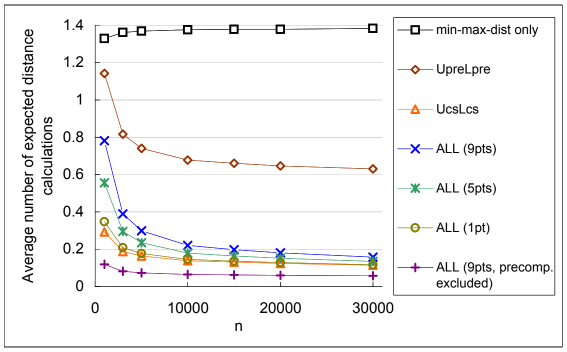


**Figure 4. $N_{ED}$ vs. $n$.**

In the figure, "ALL" refers to the use of all four pruning methods, and the bracketed number refers to the number of anchor points used in $U_{pre}$ and $L_{pre}$. The nine-point scheme was used by default if not otherwise specified. All curves that involve $U_{pre}$ or $L_{pre}$ except the one labeled "All (9pts, precomp. excluded)" include the overhead of pre-computing the expected distances for the anchor points.

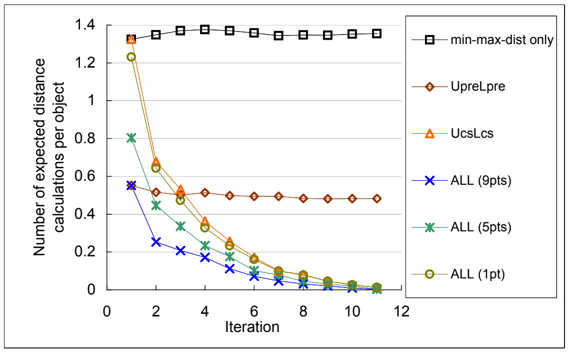
Recall that $k = 49$ in this experiment (baseline setting, see Table 6.1). Therefore, the brute-force UK-means algorithm computes 49 expected distances per object per iteration. From Figure 4, we see that min-max-dist pruning reduces $N_{ED}$ to slightly less than 1.4, i.e., about 97% of the expected distance computations are pruned.

Using 9-point pre-computation for $U_{pre}L_{pre}$ (the '$\diamond$' line) further improves the effectiveness of min-max-dist pruning by a factor of 2 when $n$ is large. Method $U_{cs}L_{cs}$ (the '$\triangle$' line), which uses bounds based on the cluster-representative-shift triangle inequality, gives the best performance. In terms of pruning effectiveness, the figure shows that $U_{cs}L_{cs}$ is 4.5 to 12 times more effective than the basic min-max-dist pruning. For example, when $n = 20,000$, $N_{ED}$ is only 0.12. Algorithms that use all four pruning methods (the "ALL" curves) perform better than $U_{pre}L_{pre}$, but are seen to be less effective than $U_{cs}L_{cs}$. This is mainly due to the pre-computation overheads. Comparing the three "ALL" curves, we see that the saving made by the use of more anchor points cannot compensate for the extra overhead induced. On the other hand, if the pre-computations are discounted from the cost model such as for those cases in which the pre-computed anchor point expected distances are reused over and over again across multiple clustering exercises, then the combination of all 4 methods (the '+' line) gives the most effective pruning. As shown in Figure 4, the pruning effectiveness is about 11 to 24 times better than basic min-max-dist. This leads to a very significant performance improvement over all others. Finally, we have also tried more than nine anchor points, but they did not give any significant further improvement. The results are thus not shown in the graph.

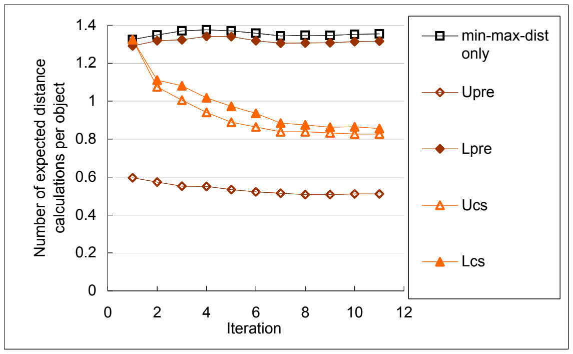
From Figure 4, we see that the four methods become more effective (i.e., their values of $N_{ED}$ become smaller) as $n$, the number of objects, increases. We note that with more objects, the number of iterations executed by the algorithms becomes larger before the clusters stabilize. A higher number of iterations favors $U_{pre}$ and $L_{pre}$ due to the amortization of the pre-computation overheads. It also favors $U_{cs}$ and $L_{cs}$ since the cluster representatives move only mildly in later iterations of the algorithm.

To further illustrate this point, we analyze the number of expected distance calculations performed in each iteration of a clustering process. Figure 5a shows a typical breakdown of the expected distance calculations across iterations for a dataset with 1,000 objects. The figure shows, for example, that using all 4 pruning methods with 5 anchor points, the algorithm computes an average of 0.8 expected distances per object during iteration 1. From Figure 5, we see that the number of expected distance calculated by basic min-max-dist and $U_{pre}L_{pre}$ stay relatively stable across the iterations. This number, however, drops rapidly across iterations when $U_{cs}$ and $L_{cs}$ are used. During the late iterations, cluster representatives shift only slightly and $U_{cs}L_{cs}$ registers a very significant pruning result.

We further study the behavior of each of the four pruning methods individually. The results are shown in Figure 5b.



(a)



(b)

**Figure 5. Number of expected distance calculations in each iteration number ($n = 1,000$).**

From the figure, we see that $L_{pre}$ offers only slightly better pruning than basic min-max-dist. Also, among the four pruning methods, when applied alone, $U_{pre}$ is the most effective. The lower bound estimated by Inequality 3 is therefore not much tighter than the minimum distance from an object's MBR to a cluster representative (see Figure 1). Comparing the curve for $U_{pre}$ in Figure 5(b) to that for $U_{pre}L_{pre}$ in Figure 5(a), we see that adding the pruning method $L_{pre}$ to $U_{pre}$ achieves only a small gain. The effect of $L_{pre}$ and $U_{pre}$ is somewhat additive to each other.

Methods $U_{cs}$ and $L_{cs}$, however, are synergetic to each other. From Figure 5(b), we see that $U_{cs}$ and $L_{cs}$ achieve similar pruning effectiveness. However, when combined, they achieve a tremendous pruning effectiveness. This is especially so towards the later iterations. The synergetic effect of $U_{cs}$ and $L_{cs}$ indicates that we need good estimations for both an upper bound and a lower bound when we apply the cluster shift method. On the other hand, for the pre-computation method, an accurate upper bound estimation is more important.

We have also studied the effectiveness of the pruning methods when the objects exhibit intrinsic cluster patterns. Figure 6 shows the results. Comparing Figure 6 and Figure 4, we observe that the general trends of the curves remain the same as in the case when data are generated without cluster patterns. We note that $N_{ED}$ values are generally smaller in Figure 6 than those in Figure 4. This is because if data objects follow certain cluster patterns, then objects tend to get drawn towards their respective clusters. As a consequence, there will be fewer objects that are "close" to multiple cluster representatives. It is thus easier to prune irrelevant representatives. Pruning effectiveness is therefore slightly higher.

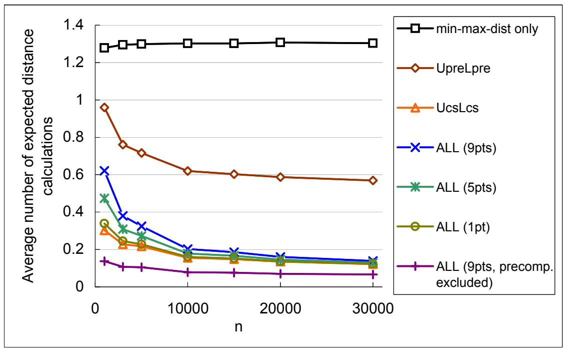


**Figure 6. $N_{ED}$ vs. $n$ (on data with cluster patterns).**

Our next experiment studies the effectiveness of the pruning methods when the number of clusters ($k$) varies from 9 to 81. Figure 7 shows the results when 20,000 objects are clustered. Since the number of expected distance calculations should increase with $k$, instead of reporting the absolute number of calculations, we report the values as a percentage of the number required by the brute-force UK-means algorithm.

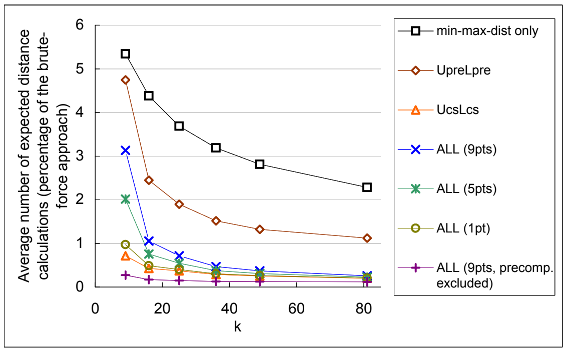


**Figure 7. Average number of expected distance calculations vs. $k$.**

From the figure, again we see that the four pruning methods significantly outperform basic min-max-dist pruning. In general, a larger $k$ gives a more effective pruning (relative to the brute-force UK-means algorithm). This is because with more clusters, there are simply more candidates to be pruned. Method $U_{pre}L_{pre}$ again is about twice as effective as the basic min-max-dist pruning. Also, without the pre-computation overheads, $U_{cs}L_{cs}$ continues to be the best strategy. If overheads are discounted, then using all 4 pruning methods gives a very impressive pruning effectiveness. The ratios between the min-max-dist curve and each of the other curves remain virtually constant for most values of $k$, except when $k$ is very small. This is because under small $k$, the overhead of $U_{pre}$ and $L_{pre}$ becomes relatively significant. Overall, the figure shows that the pruning methods are extremely effective over a broad range of values of $k$.

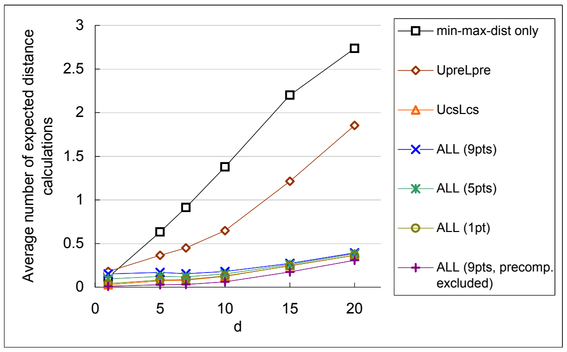
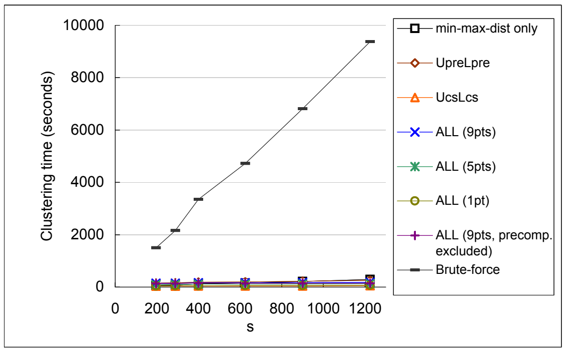


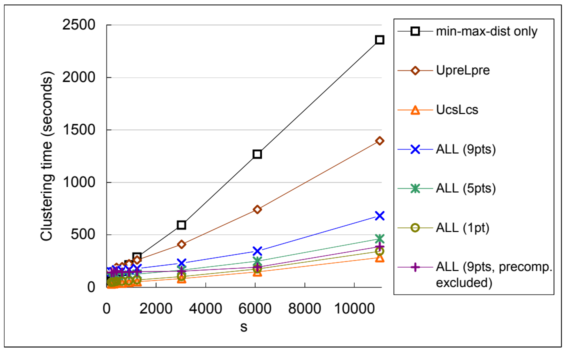
**Figure 8. $N_{ED}$ vs. $d$.**

The next experiment studies the effect of $d$, the maximum length of an MBR's edge. Figure 8 shows how $N_{ED}$ changes as $d$ varies from 1 to 20 units. From the figure, we see that all curves rise with $d$. This is because a larger $d$ gives larger uncertainty regions of objects, which leads to more data uncertainty. As we have explained through the illustration shown in Figure 1, a bigger bounding box results in less effective pruning, which is reflected by the curves in Figure 8. An interesting observation from the figure is that when $d$ is very small, $U_{pre}$ and $L_{pre}$ are not more effective than the basic min-max-dist pruning algorithm. This is because with a very small MBR, an expected distance estimated using an anchor point is not much different from those estimated using the minimum and maximum distances to the MBR. In comparison, $U_{cs}$ and $L_{cs}$ are relatively more effective. Except for extremely small values of $d$, all combinations of the four pruning methods are highly effective and they provide substantial improvement to the basic min-max-dist pruning.

Finally we study the relationship between the actual running time and the number of sample points $s$ used to represent the pdf of an object. Recall that a larger $s$ implies a more expensive expected distance calculation. So, for large $s$ values, we expect that the more effective is a pruning method, the faster is the algorithm. We use a dataset with 10,000 objects without cluster patterns for this study. We vary $s$ from 196 to 1,225. The average results of 10 clustering runs are shown in Figure 9.



(a) including brute-force UK-means.



(b) pruning methods only.

**Figure 9. Time vs. $s$.**

Figure 9a compares the execution times of all approaches, including that of the brute-force UK-means algorithm (which does not perform any pruning). It is clear that for all values of $s$, a substantial amount of time is saved by the pruning methods. Even for the smallest value of $s$ (196), applying any kinds of pruning still reduces the running time by at least a factor of ten. We remark that in practice, $s$ should be much larger than what we have shown in the graph (such as in the order of $10^4 - 10^5$) such that the pdf's are accurately represented. Hence, brute-force UK-means is not feasible computationally.

Figure 9b shows only the curves that involve pruning. From the figure, we see that all curves go up with $s$. This is because a larger $s$ implies a higher computational cost in calculating an expected distance. Among all the curves, min-max-dist pruning rises most rapidly with $s$. This is because min-max-dist is the least effective in pruning and therefore it is the most sensitive to $s$. If $s$ is extremely small (e.g., 196), computing expected distances are not that expensive and min-max-dist performs very well. The pre-computation-based methods give no performance gain due to the various overheads involved. Therefore using all four pruning methods does not run faster than using basic min-max-dist pruning only. Yet, $U_{cs}L_{cs}$ and $ALL(1pt)$ are still amongst the best strategies, as they save expected distance calculations while not introducing too much computational overhead.

When $s$ is moderate (e.g., 3,025), the cost of expected distance computation is major and it dominates the algorithms' execution times. The basic min-max-dist pruning method is outperformed by any combination of the four pruning methods. The relative performance of the different methods now follow almost the same trend as that shown in Figure 4.

## 7. Concluding remarks

In this paper we studied the problem of clustering uncertain objects with the uncertainty regions defined by pdfs. For an accurate representation, at least thousands of sample points should be used to approximate an object's pdf. When applying the UK-means algorithm to cluster uncertain objects, a large number of expected distances have to be calculated. We explained why expected distance computations are expensive and thus argued that effective pruning techniques are necessary for a computationally feasible clustering algorithm.

We described the basic min-max-dist pruning method and showed that it was fairly effective in pruning expected distance computations. To further improve performance, we derived four bound-estimation methods. We conducted extensive experimental study evaluating those four pruning methods. Our results showed that $U_{cs}$ and $L_{cs}$ are very effective, especially when they work together. In some experiment setting, $U_{cs}L_{cs}$ was a dozen times more effective than basic min-max-dist in terms of pruning effectiveness. Method $U_{pre}L_{pre}$, which is based on pre-computation of anchor points' expected distances, also performed very well. The pre-computation overheads, however, made $U_{pre}L_{pre}$ second-best to $U_{cs}$ and $L_{cs}$. The four pruning methods are independent of each other and can be combined to achieve an even higher pruning effectiveness. Pruning is at its full-strength when all four are applied and if the pre-computation overhead could be discounted. A factor of 24 times more effective in pruning than min-max-dist was registered in some of the experiments.

## References

[1] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. In *Proc. of ACM SIGMOD Conference*, 1999.

[2] M. Chau, R. Cheng, B. Kao, and J. Ng. Uncertain data mining: An example in clustering location data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2005.

[3] R. Cheng, D. Kalashnikov, and S. Prabhakar. Querying imprecise data in moving object environments. *IEEE TKDE*, 16(9):1112–1127, 2004.

[4] R. Cheng, X. Xia, S. Prabhakar, R. Shah, and J. Vitter. Efficient indexing methods for probabilistic threshold queries over uncertain data. In *Proc. of VLDB Conference*, 2004.

[5] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood for incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[6] J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3:32–57, 1973.

[7] C. Elkan. Using the triangle inequality to accelerate k-means. In *Proc. of ICML Conference*, 2003.

[8] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of ACM SIGKDD Conference*, 1996.

[9] H. Hamdan and G. Govaert. Mixture model clustering of uncertain data. In *Proc. of IEEE ICFS Conference*, pages 879–884, 2005.

[10] E. Hung, L. Getoor, and V. S. Subrahmanian. PXML: A probabilistic semistructured data model and algebra. In *Proc. of IEEE ICDE Conference*, 2003.

[11] M. Ichino and H. Yaguchi. Generalized minkowski metrics for mixed feature type data analysis. *IEEE TSMC*, 24(4):698V–708, 1994.

[12] H.-P. Kriegel and M. Pfeifle. Density-based clustering of uncertain data. In *Proc. of ACM SIGKDD Conference*, 2005.

[13] H.-P. Kriegel and M. Pfeifle. Hierarchical density-based clustering of uncertain data. In *Proc. of IEEE ICDM Conference*, 2005.

[14] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[15] M. Nanni. Speeding-up hierarchical agglomerative clustering in presence of expensive metrics. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 378–387, 2005.

[16] N. D. Nilesh and D. Suciu. Efficient query evaluation on probabilistic databases. In *Proc. of VLDB Conference*, pages 864–875, 2004.

[17] E. H. Ruspini. A new approach to clustering. *Information Control*, 15(1):22–32, 1969.

[18] M. Sato, Y. Sato, and L. Jain. *Fuzzy Clustering Models and Applications*. Physica-Verlag, Heidelberg, 1997.

[19] O. Wolfson, P. Sistla, S. Chamberlain, and Y. Yesha. Updating and querying databases that track mobile units. *Distributed and Parallel Databases*, 7(3), 1999.